Université de Montréal

Métaheuristiques de recherche avec tabous pour le problème de synthèse de réseau multiproduits avec capacités

par

Ilfat Ghamlouche
Département d'informatique et de recherche opérationnelle
Faculté des arts et des sciences

Thèse présentée à la Faculté des études supérieures
en vue de l'obtention du grade de
Philosophiæ Doctor (Ph.D.)
en informatique et recherche opérationnelle

Février, 2004

©Ilfat Ghamlouche, 2004

Faculté des études supérieures
Grade conféré à compter du
2004 MAI 06
Université de Montréal

QA
76
U5U
2004
.1.007

Université de Montréal

Direction des bibliothèques

AVIS

L'auteur a autorisé l'Université de Montréal à reproduire et diffuser, en totalité ou en partie, par quelque moyen que ce soit et sur quelque support que ce soit, et exclusivement à des fins non lucratives d'enseignement et de recherche, des copies de ce mémoire ou de cette thèse.

L'auteur et les coauteurs le cas échéant conservent la propriété du droit d'auteur et des droits moraux qui protègent ce document. Ni la thèse ou le mémoire, ni des extraits substantiels de ce document, ne doivent être imprimés ou autrement reproduits sans l'autorisation de l'auteur.

Afin de se conformer à la Loi canadienne sur la protection des renseignements personnels, quelques formulaires secondaires, coordonnées ou signatures intégrées au texte ont pu être enlevés de ce document. Bien que cela ait pu affecter la pagination, il n'y a aucun contenu manquant.

NOTICE

The author of this thesis or dissertation has granted a nonexclusive license allowing Université de Montréal to reproduce and publish the document, in part or in whole, and in any format, solely for noncommercial educational and research purposes.

The author and co-authors if applicable retain copyright ownership and moral rights in this document. Neither the whole thesis or dissertation, nor substantial extracts from it, may be printed or otherwise reproduced without the author's permission.

In compliance with the Canadian Privacy Act some supporting forms, contact information or signatures may have been removed from the document. While this may affect the document page count, it does not represent any loss of content from the document.

Université de Montréal

Faculté des études supérieures

Cette thèse intitulée :

"Métaheuristiques de recherche avec tabous pour le problème de synthèse de réseau multiproduits avec capacités"

présentée par :

Ilfat Ghamlouche

a été évaluée par un jury composé des personnes suivantes :

Jean-Yves Potvin	président du jury
Michel Gendreau	directeur de recherche
Teodor Gabriel Crainic	codirecteur de recherche
Bernard Gendron	membre du jury
Cesar Rego	examinateur externe
Guy Desaulniers	représentant du doyen

Thèse acceptée le :

Sommaire

Le problème de synthèse de réseau multiproduits avec capacités représente un modèle générique pour une grande variété d'applications incluant la planification, le développement, l'amélioration, et l'opération des systèmes du transport, la logistique, les télécommunications et les systèmes production-distribution.

Dans cette thèse, nous sommes intéressés tout particulièrement à l'élaboration de métaheuristiques de recherche avec tabous pour produire des solutions de bonne qualité pour ce problème. Nos principales contributions se résument comme suit. Dans le chapitre 2, nous présentons une description des élements de base de la méthode de recherche avec tabous et de certaines de ses stratégies de recherche. Nous y passons également une revue des réalisations les plus pertinentes pour le problème de synthèse de réseau multiproduits avec capacités.

Dans le chapitre 3, nous proposons une nouvelle structure de voisinage pour les métaheuristiques dédiées au problème de synthèse de réseau multiproduits avec capacités. Cette structure de voisinage est basée sur des cycles qui seront détectés et évalués dans des graphes résiduels. L'idée fondamentale est de définir des mouvements qui affectent plusieurs produits en même temps et qui ouvrent et ferment plusieurs arcs à la fois. Le mouvement consiste donc dans la déviation du flot total d'un chemin à un autre. Ces deux chemins relient deux noeuds quelconques du réseau (ne sont pas restreints à relier l'origine et la destination d'un produit) et forment un cycle. La déviation du flot autour de ce cycle résultera dans l'ouverture et la fermeture de certains arcs du cycle. Nous développons une méthode simple de recherche avec tabous pour tester cette structure de voisinage. Les résultats obtenus indiquent clairement la supériorité de cette approche par rapport à la meilleure heuristique proposée dans la littérature.

Dans le chapitre 4, nous proposons une procédure de "path relinking" pour notre problème. Des solutions élites sont construites en exécutant un algorithme de recherche avec tabous qui utilise des structures de voisins basées sur des cycles présenté dans le chapitre 3. Une version restreinte de la même structure de voisinage est utilisée pour explorer le chemin entre deux solutions élites candidates. Nous développons et

comparons plusieurs variantes de la méthode. Les résultats numériques indiquent que cette méthode constitue la meilleure métaheuristique actuellement disponible pour cette classe de problèmes difficiles.

Dans le chapitre 5, nous proposons une méthode basée sur des mécanismes d'apprentissage pour aborder notre problème. De ce fait, nous développons et examinons des procédures d'intensification et de diversification utilisant des structures de mémoires adaptatives. Les résultats obtenus indiquent que l'algorithme proposé est efficace pour résoudre les problèmes de grande taille avec plusieurs produits.

Mots-clés : Synthèse de réseau multiproduits avec capacités, Métaheuristiques, Voisinages par cycles, Méthodes de recherche avec tabous, Chemins reliants, Mémoires adaptatives.

Abstract

The fixed-charge capacitated multicommodity network design formulation represents a generic model for a wide range of applications in planning, development, improvement, and operation of transportation systems, logistics, telecommunication, and production-distribution systems.

In this thesis, we are particularly interested in developing meta-heuristics with tabu search to produce high quality solutions to this problem. Our main contributions can be summarized as follows. In chapter 2, we present a description of the basic elements of the tabu search method and associated search strategies. We also present an overview of relevant contributions addressing the fixed-charge capacitated multicommodity network design problem.

In chapter 3, we propose a new class of neighbourhood structures for meta-heuristics for the fixed-charge capacitated multicommodity network design problem. The new neighbourhood structures are based on cycles detected and evaluated on residual networks. The fundamental idea at the origin of the new neighbourhood class is that one may move from one solution to another by : 1) identifying two points in the network together with two paths connecting these points, thus forming a cycle; 2) deviating the total flow from one path to another such that at least one currently open arc becomes empty; 3) closing all previously open arcs in the cycle that are empty following the flow deviation and, symmetrically, opening all previously closed arcs that now have flow. To illustrate the quality of this new neighbourhood, we develop a simple tabu search method. Experimental results show clearly the superiority of the proposed approach over the best heuristic reported in the literature.

In chapter 4, we propose a path relinking procedure for the fixed-charge capacitated multicommodity network design problem. Cycle-based neighbourhoods are used both to move along paths between elite solutions and to generate the elite candidate set by a tabu-like local search procedure. Several variants of the method are implemented and compared. Numerical experiments indicate that the method constitute

the best current meta-heuristic for this difficult class of problems.

In chapter 5, we propose a method based on learning mechanisms to address our problem. We develop and examine concepts widely used in combinatorial Tabu search, such as intensification and diversification mechanisms. Experimental results show that the proposed algorithm is effective for large structured instances with several commodities.

Keywords : Fixed charge capacitated multicommodity network design, Meta-heuristics, Cycle-based neighbourhoods, Tabu search, Path relinking, Adaptive memories

Table des matières

Table des figures

Liste des tableaux

À mon époux Samih.

Remerciements

Je désire exprimer ma profonde gratitude à mes directeurs, les professeurs Michel Gendreau et Teodor Gabriel Crainic de m'avoir donné la chance d'accomplir cette thèse sous leur direction. Je suis également reconnaissante envers eux pour leurs encouragements continus, leur implication remarquable, leur soutien financier, ainsi que la confiance qu'ils m'ont témoignée tout au long de ces années de travail conjoint. J'espère sincèrement que cette thèse n'est que le début de collaborations futures tout aussi fructueuses.

Je remercie aussi les professeurs, chercheurs et étudiants que j'ai connus au Centre de recherche sur les transports et au Département d'informatique et de recherche opérationnelle pour avoir su créer une atmosphère aussi plaisante qu'enrichissante. Je remercie également les responsables des ressources informatiques Daniel Charbonneau, Luc Rocheleau, François Guertin et tout particulièrement Pierre Girard pour l'aide qu'ils m'ont apportée tout au long de ces années d'études.

Je remercie tous mes ami(e)s pour leur soutien moral et leur encouragement. Je remercie tout particulièrement Mervat Chouman pour les beaux moments qu'on a eus ensemble durant ces longues dernières années.

Finalement, tout mon respect et ma gratitude vont à mon époux Samih pour sa présence, son soutien, son encouragement et ses conseils, et sans qui cette thèse n'aurait pu être complétée.

Chapitre 1

Introduction

Le problème de synthèse de réseau multiproduits avec capacités est un sous-problème d'une large famille connue dans la littérature sous le nom "problèmes de synthèse de réseau" (Minoux 1989, Magnanti et Wong 1984). Cette classe englobe entre d'autres des problèmes de transport, de télécommunication, de logistique, de production-distribution et recouvre également tous les niveaux dans la planification des entreprises, le stratégique, le tactique et l'opérationnel (Crainic et Laporte 1997). Nous donnons trois exemples illustrant chacun un niveau de planification différent.

Le niveau stratégique traite les décisions budgétaires à prendre pour une planification à long terme. Dans l'évolution du réseau routier, il est toujours nécessaire de planifier des constructions futures pour répondre à la croissance continue de la population. Déterminer les emplacements et les budgets associés aux constructions futures sont des décisions à prendre au niveau stratégique. Le niveau tactique détermine sur un horizon de planification à moyen terme, la façon la plus efficace, économique et rentable pour l'utilisation des ressources disponibles. A ce niveau, nous disposons généralement d'un ensemble de ressources, d'un réseau et d'une demande. Le problème consiste à utiliser les ressources disponibles pour satisfaire la demande à travers le réseau. Au niveau tactique, un problème de transport urbain, revient à sélectionner parmi les lignes possibles (origines et destinations, arrêts intermédiaires) celles qui peuvent satisfaire la demande à moindre coût. Finalement, les problèmes de synthèse de réseau sont aussi un outil pour une planification à court terme. Pour illustrer une situation exigeant un tel horizon de planification, nous pouvons prendre l'exemple de l'affectation journalière des ressources disponibles. Relier entre eux les voyages de

différentes lignes de façon à optimiser le nombre d'autobus nécessaires est un exemple de ce niveau de planification.

Dans cette thèse, nous nous intéressons à la formulation suivante : étant donné un réseau orienté, caractérisé par des arcs possédant des coûts et des capacités, et un ensemble de produits à acheminer entre différentes paires d'origines et de destinations de ce réseau, notre objectif consiste à sélectionner les arcs à être utilisés pour répondre aux exigences de demandes et de capacités. Cette sélection doit être optimale en fonction des coûts fixes à payer pour utiliser les arcs et des coûts de transport associés à chaque produit.

Bien que de nombreux problèmes de la vie pratique puissent être modélisés comme un problème de synthèse de réseau multiproduits avec capacités, peu d'algorithmes existent pour sa résolution (Balakrishnan, Magnanti et Mirchandani 1997). La présence des contraintes de capacités, des caractéristiques combinatoires, telles les exigences d'intégralité de certaines variables, ainsi que la grande taille des applications réelles compliquent singulièrement le problème. La présence du coût fixe et du coût variable dans la fonction objective nous empêche de construire le réseau en minimisant les coûts fixes pour ensuite résoudre un problème de multiflot à coût minimum. Elle nous empêche également de résoudre un problème de multiflot en minimisant les coûts variables pour ensuite fermer les arcs non utilisés. Notre problème est considéré comme étant NP-complet. Toutefois, étant donné la complexité de ce problème, seulement des instances de petite taille peuvent être résolus efficacement en temps raisonnable. La plupart des chercheurs se sont contentés de présenter le problème et de donner des pistes potentielles pour sa résolution. Des travaux sont réalisés pour étudier des cas particuliers : sans capacité, sans coût variable, etc. La généralisation des algorithmes développés pour résoudre les problèmes de synthèse de réseau sans capacité étaient sans succès pour résoudre les problèmes avec capacités (Balakrishnan, Magnanti et Mirchandani 1997) qui sont beaucoup plus complexes.

Face à ces défis, cette thèse trouve sa motivation dans la volonté de résoudre efficacement le problème de synthèse de réseau multiproduits avec capacités. Nous visons particulièrement à produire des solutions de bonne qualité pour ce problème combinatoire extrêmement difficile.

Les heuristiques sont souvent proposées pour résoudre les problèmes complexes d'optimisation combinatoire. Dans le cadre de cette thèse, nous choisissons la recherche avec tabous parce qu'elle offre des avenues extrêmement intéressantes pour obtenir des solutions de très bonne qualité à des problèmes complexes dans divers contextes : les problèmes d'horaire (Mooney et Rardin 1992), de transport (Gendreau, Hertz et Laporte 1994, Crainic et al. 1993), de télécommunication (Berger et al. 2000), etc...

Outre le présent chapitre, cette thèse se compose de quatre chapitres, chacun pouvant être lu indépendamment des autres. De ce fait, la bibliographie relative à chacun d'eux, se trouve à la fin du chapitre. Dans le chapitre 2, nous présentons les éléments de base de la méthode de recherche avec tabous et certaines de ses stratégies de recherche. Nous présentons également la formulation du problème de synthèse de réseau multiproduits avec capacités, nous spécifions la problématique associée, nous décrivons des méthodes de résolution et nous passons en revue les principaux travaux portant sur des méthodes heuristiques et des relaxations.

Les trois chapitres suivants se concentrent de façon plus spécifique sur la résolution du problème. Le chapitre 3 correspond à un article publié dans *Operations Research*, le chapitre 4 correspond à un article qui est à paraître dans *Annals of Operations Research* et le chapitre 5 sera soumis dans un très proche délai à une revue scientifique.

Dans le chapitre 3, nous proposons une nouvelle classe de voisinages pour le problème de synthèse de réseau multiproduits avec capacités. Les mouvements ainsi définis prennent explicitement en compte l'impact sur le coût du design de modifications simultanées à la distribution de plusieurs produits. L'identification de bons mouvements se fait par l'intermédiaire d'une procédure d'optimisation basée sur une méthode de calcul de chemins les plus courts. Une solution voisine est obtenue par la re-direction de flots dans des cycles et la fermeture/ouverture subséquente d'arcs de design. La nouvelle classe de voisinages est testée dans le contexte d'une simple recherche avec tabous et les résultats numériques indiquent clairement la supériorité de cette approche par rapport à la meilleure heuristique proposée dans la littérature.

Dans le chapitre 4, nous proposons une procédure de "path relinking" pour notre problème. Nous visons, en particulier à assurer une exploration plus large du domaine

de solutions en explorant des chemins qui lient des solutions élites. Des mouvements basés sur des cycles (vu au chapitre 3) sont utilisés autant pour explorer les chemins entre les solutions élites, que pour générer l'ensemble des solutions élites candidates. Nous développons et comparons plusieurs variantes de la méthode. Les résultats numériques indiquent que la procédure proposée permet d'améliorer la robustesse des solutions produites au chapitre 3.

Dans le chapitre 5, nous abordons et analysons des stratégies de mémoires adaptatives proposées au sein de la méthode de recherche avec tabous. Nous proposons des procédures d'intensification et de diversification pour notre problème. Nous examinons particulièrement la manière d'utiliser l'information passée pour éviter des situations où la recherche se cantonne dans une portion peu intéressante du domaine de solutions. Les résultats numériques indiquent que les procédures proposées sont efficaces pour résoudre les problèmes de grande taille avec plusieurs produits.

Finalement, dans le chapitre 6 nous soulignons les principales contributions de cette thèse et nous proposons quelques avenues de recherche qu'il serait pertinent d'explorer.

Bibliographie

Balakrishnan, A., Magnanti, T.L., and Mirchandani, P. (1997). Network Design. In Dell'Amico, M., Maffioli, F., and Martello, S., editors, *Annotated Bibliographies in Combinatorial Optimization*, pages 311–334. John Wiley & Sons, New York, NY.

Berger, D., Gendron, B., Potvin, J.-Y., Raghavan, S., and Soriano, P. (2000). Tabu Search for a Network Loading Problem with Multiple Facilities. *Journal of Heuristics*, 6(2) :253–267.

Crainic, T.G., Gendreau, M., Soriano, P., and Toulouse, M. (1993). A Tabu Search Procedure for Multicommodity Location/Allocation with Balancing Requirements. *Annals of Operations Research*, 41 :359–383.

Crainic, T.G. and Laporte, G. (1997). Planning Models for Freight Transportation. *European Journal of Operational Research*, 97(3) :409–438.

Gendreau, M., Hertz, A., and Laporte, G. (1994). A Tabu Search Heuristic for the Vehicle Routing Problem. *Management Science*, 40 :1276–1290.

Magnanti, T.L. and Wong, R.T. (1984). Network Design and Transportation Planning : Models and Algorithms. *Transportation Science*, 18(1) :1–55.

Minoux, M. (1989). Network Synthesis and Optimum Network Design Problems : Models, Solution Methods and Applications. *Networks*, 19 :313–360.

Mooney, E.L. and Rardin R.L. (1992). Tabu Search for a Class of Scheduling Problems. *Annals of Operations Research*, 41 :253–278.

Chapitre 2

Méthode de recherche avec tabous et problème de synthèse de réseau multiproduits avec capacités

Dans ce chapitre, nous présentons en premier la méthode de recherche avec tabous, nous rappelons ses éléments de base, puis nous décrivons certaines de ses stratégies de recherche. Nous présentons ensuite la formulation associée au problème de synthèse de réseau multiproduits avec capacités, nous décrivons les problèmes apparentés pertinents à notre étude, puis nous abordons les méthodologies générales adaptées pour sa résolution.

2.1 Recherche avec tabous

La recherche avec tabous est une métaheuristique introduite par Glover en 1986. Elle est basée sur un principe d'intelligence artificielle pour résoudre les problèmes d'optimisation (Glover et Laguna 1997, Glover 1989, Glover 1990, Glover 1996). Elle combine une méthode de recherche locale basée sur une définition de voisinage de la solution courante et des mémoires. Le rôle de ces mémoires consiste à garder la trace du cheminement passé du processus pour guider l'exploration.

2.1.1 Fondement et éléments de bases

Durant plusieurs années, des efforts ont été accomplis dans l'exploration des stratégies de recherche locale pour les problèmes d'optimisation combinatoire de la

forme :

$$\begin{array}{ll} \min & f(x) \\ sujet\ à & x \in S \end{array}$$

où $f(x)$ est une fonction non restreinte à être linéaire, et S est un ensemble de contraintes défini sur x. Ce dernier peut avoir des composantes entières ou mixtes.

De nombreuses heuristiques ont été développées dans ce contexte, la plus connue étant certainement la méthode de descente. Cette méthode est caractérisée par le fait qu'on se déplace d'une solution x à une autre, y, dans son voisinage pour réduire la valeur de la fonction objective. Un voisin de x est une solution obtenue à partir de x par un certain mécanisme, qui peut être défini de différentes façons, et le voisinage $N(x)$ est défini comme étant :

$$N(x) = \{y \in S : y \text{ est un voisin de } x\}. \tag{2.1}$$

Le principal inconvénient de cette méthode est de s'arrêter au premier optimum local rencontré (i.e x tel que $f(y) \geq f(x)\ \forall\ y \in N(x)$), ce dernier pouvant être très loin de l'optimum global (i.e x tel que $f(y) \geq f(x)\ \forall\ y \in S$). L'algorithme de la méthode de descente est résumé à la figure 2.1.

x= solution de départ
min $\leftarrow \infty$
stop = faux
répéter
    choisir x^* tel que $f(x^*) = \min\{f(y) : y \in N(x)\}$
    si $f(x^*) < f(x)$ alors $x = x^*$
    Sinon stop = vrai
jusqu'à stop
x est un minimum local

FIG. 2.1 – Algorithme de descente

Pour éviter de s'arrêter à un minimum local, Glover a introduit une nouvelle méthode de recherche appelée recherche avec tabous. Cette méthode est célèbre pour sa performance et la qualité des solutions engendrées. Elle a déjà démontré son potentiel pour aborder les problèmes d'optimisation combinatoire de grande taille.

Cependant, l'efficacité et la qualité d'un algorithme de recherche avec tabous peut être grandement affecté par la modélisation du problème, c'est-à-dire par le choix de l'espace de solutions à explorer, le voisinage $N(x)$ et la fonction objective. Ces choix ont une importance non négligeable. Il est indispensable de définir l'espace de solutions, de s'assurer que la structure de voisinage doit permettre de se déplacer facilement d'une solution à une autre et qu'elle garantisse qu'à partir d'une solution donnée x, il existe toujours un chemin menant de x à une solution optimale. La fonction objective devrait fournir assez d'information pour permettre de départager facilement les solutions candidates à chaque itération. Donc, ces choix sont primordiaux et doivent se baser sur une analyse sérieuse du problème à résoudre.

La recherche avec tabous est itérative ; elle consiste à générer une suite de solutions à partir d'une solution initiale. A chaque itération, on détermine le voisinage de la solution courante et on choisit la meilleure solution contenue dans ce dernier. Pratiquement, on définit un voisin de x (ou un élément du voisinage de x) comme une solution admissible obtenue à partir de x selon un mécanisme spécifique au problème traité. Le meilleur voisin de x n'entraîne pas nécessairement une réduction de la fonction objective. Lorsque la solution retenue n'entraîne pas une réduction (descente), elle représente la plus faible détérioration dans le voisinage de la solution courante. Ainsi, si on se retrouve dans un minimum local, la recherche avec tabous permet de s'en éloigner en détériorant le moins possible la valeur de la fonction objective.

L'abscence de monotonicité dans la suite des valeurs de la fonction objective associées aux solutions pourrait provoquer du cyclage. L'idée de base de la recherche avec tabous consiste à garder le cheminement passé du processus dans une ou des mémoires et de se servir de cette information pour se rappeler du passé immédiat et éviter de revenir en arrière et cycler. En d'autres termes, la recherche conserve à chaque itération une liste de solutions taboues vers lesquelles il est interdit de se déplacer au cours d'un nombre fixé de prochaines itérations. Mais il n'est pas raisonnable de considérer les solutions comme éléments de la liste taboue puisque ce genre de liste peut rapidement s'avérer très encombrante et coûteuse à gérer étant donné la quantité d'informations requises pour décrire complètement une solution. Par conséquent, les chercheurs ont choisi d'affecter des attributs aux différentes com-

posantes d'une solution pour leur affecter le statut tabou. Lors du passage d'une solution à une autre voisine, les composantes dont les valeurs ont changé dans la nouvelle solution sont affectées d'un attribut tabou pour un nombre fixé de prochaines itérations. Dans le cas d'un graphe orienté où le passage d'une solution à une autre voisine consiste à fermer ou à ouvrir plusieurs arcs, on affecte à chaque arc du graphe l'attribut $ArcestTabou$ initialisé à 0. Durant le processus, les attributs des arcs dont les valeurs sont modifiées d'une solution à une autre, changent pour le numéro d'itération actuel + un nombre fixe k. Le test suivant nous indique si un arc possède le statut tabou :

$$\text{Nombre d'itérations} \leq ArcestTabou.$$

Un élément essentiel dans la recherche est le choix du nombre d'itérations k pour lequel un attribut possède le statut tabou. Glover et Laguna (1997) mentionnent qu'aucune valeur de k n'est montrée efficace pour toutes les classes de problèmes. Les expériences numériques ont montré que dans la plupart des cas, la valeur de k est plutôt fonction de la taille du problème. En général, une petite valeur permet l'exploration de solutions proches de l'optimum local alors qu'une grande valeur permet de s'éloigner et de se libérer de cet optimum. Il est alors convenable de faire varier la valeur de k pour produire un équilibre entre examiner une région de plus près et se déplacer dans les différentes régions de l'espace de solutions.

L'affectation des attributs tabous aux différentes composantes d'une solution rend plus facile la gestion des tabous mais peut réduire l'ensemble de transitions possibles, en interdisant des solutions non visitées, au point d'empêcher la méthode de trouver de bonnes solutions. Pour illustrer le cas de solutions interdites non visitées, prenons l'exemple d'un graphe où, à une itération donnée, l'arc (a, b) est fermé et l'arc (b, c) est ouvert et supposons qu'à l'itération suivante ces deux arcs sont complémentés : l'arc (a, b) ne peut être fermé et l'arc (c, d) ne peut être ouvert pour k itérations. Durant un nombre d'itérations égal à k, le statut tabou de ces arcs nous empêche de considérer les solutions non encore visitées où les deux arcs sont dans la même situation (ouverts ou fermés en même temps).

Pour remédier à cela, la recherche avec tabous fait intervenir un autre élément qui

s'appelle la fonction d'aspiration. Le rôle de cette fonction est de révoquer le statut tabou d'une solution si elle est jugée avantageuse. Plusieurs fonctions d'aspiration existent. La plus simple consiste à révoquer le critère tabou si cela permet d'obtenir une solution supérieure à la meilleure solution rencontrée jusqu'à présent.

Pour définir une condition d'arrêt, on se donne en général un nombre maximum d'itérations entre deux améliorations de la meilleure solution rencontrée. Dans certains cas, il est possible de déterminer une borne inférieure f^* de la fonction objective et on peut alors stopper la recherche lorsqu'on a atteint une solution x de valeur $f(x)$ proche de f^*.

2.1.2 Stratégies de recherche

Nous avons vu jusqu'à présent une utilisation à court terme de la mémoire pour éviter le processus de retourner vers des solutions déjà rencontrées. Or, le rôle des mémoires dans la recherche avec tabous va au delà de l'utilisation à court terme et influence également le processus de recherche à moyen et long terme. L'idée principale de l'utilisation des mémoires est de développer des méthodes heuristiques de fouille qui reproduisent certaines caractéristiques de la pensée humaine. Glover et Laguna (1997) étudient cette liaison et identifient des stratégies de recherche avancées comme les stratégies d'intensification, de diversification et la stratégie des chemins reliants ou "path relinking". Ces stratégies sont des concepts très importants qui viennent élargir la gamme des outils disponibles pour contrôler et guider intelligemment la recherche de bonnes solutions.

Stratégie d'intensification

Pour certains problèmes, le voisinage $N(x)$ de la solution courante x est de très grande taille et le seul moyen de déterminer la solution x' minimisant la fonction objective f sur $N(x)$ est de passer en revue l'ensemble $N(x)$ tout entier. Dans ce cas, on préfère alors générer un sous-ensemble $N^* \subseteq N(x)$ ne contenant qu'un échantilllon de solutions voisines à x, et on choisit la solution $x' \in N^*$ de valeur $f(x')$ minimale. Cette façon de choisir N^* peut entraîner le risque de perdre de bonnes solutions. Dans la figure 2.2, le fait de générer un sous-ensemble de voisins empêche de considérer la

solution m. Donc, ce minimum ne sera jamais visité.

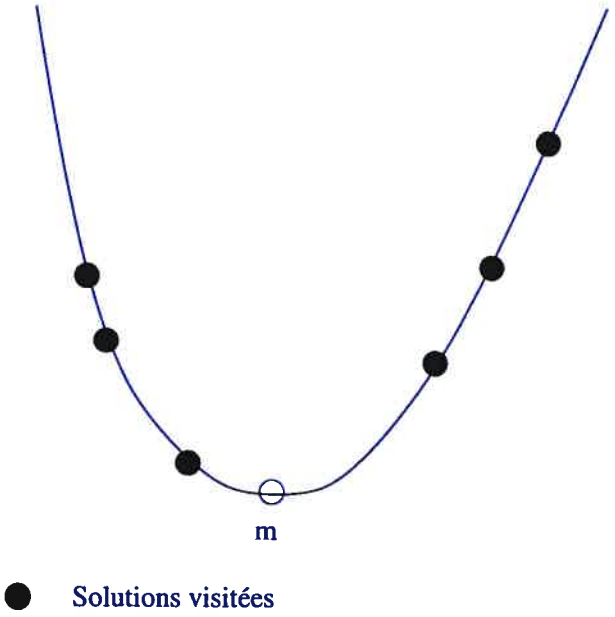


FIG. 2.2 – Intensification

Un des aspects de la stratégie d'intensification consiste alors à retourner vers des régions intéressantes pour les fouiller davantage. Un exemple simple est illustré dans la figure 2.3 et consiste à enregistrer les solutions considérées bonnes durant la recherche, puis à partir de ces solutions, essayer de générer des solutions meilleures soit en élargissant l'ensemble du voisinage, soit en modifiant le choix du sous-ensemble de voisinage déjà considéré.

Deux stratégies peuvent être utilisées pour la sélection de bonnes solutions. La première consiste à enregistrer la meilleure solution rencontrée durant chaque descente. Le choix aléatoire du sous-ensemble de voisinage nous assure de ne pas répéter la même séquence de mouvements à partir des meilleures solutions. La seconde stratégie garde une liste séquentielle d'une longueur donnée et ajoute une nouvelle solution à la fin de la liste si elle est meilleure que les solutions déjà trouvées. La mémoire à court terme associée à chacune de ces solutions sera aussi sauvegardée pour éviter au système de répéter la même exploration. Lors de l'intensification, la recherche est reprise à partir de la solution placée à la fin de la liste.

A moyen terme, la recherche avec tabous est capable d'identifier certaines caractéristiques communes aux bonnes solutions. Un autre aspect de la stratégie d'in-

Appliquer un tabou simple avec une mémoire à court terme
pour générer des solutions
Appliquer une stratégie de sélection de bonnes solutions,
i.e garder une liste $\mathcal{L}$ de k bonnes solutions ($k = 5$ à 30)
Pour chacune des solutions gardées, enregistrer les mémoires à court terme
associées
Tant que ($\mathcal{L} \neq \phi$) faire
    Choisir une solution s de la liste
    Faire $\mathcal{L} = \mathcal{L} \backslash s$
    Reprendre la recherche à partir de s pour un nombre i d'itérations
    Si les nouvelles solutions trouvées satisfont la stratégie de sélection de
    bonnes solutions
        Si $|\mathcal{L}| < k$, ajouter les à $\mathcal{L}$.
        Sinon ajouter les à $\mathcal{L}$ pour remplacer les solutions de moindre qualités

FIG. 2.3 – Approche d'intensification

tensification consiste à fixer ces caractéristiques dans la structure d'une solution et de se concentrer pour déterminer les autres parties de cette structure. Ce type d'intensification a prouvé sa puissance dans les problèmes mixtes d'optimisation combinatoire qui contiennent des variables binaires.

Stratégie de diversification

Dans les méthodes de recherche locale, la diversification est utilisée pour éviter à la recherche de répéter la même séquence de mouvements. Habituellement, nous débutons la recherche à partir de différentes solutions initiales et nous espérons trouver un minimum global. Dans la recherche avec tabous, la stratégie de diversification n'est pas un mécanisme aveugle qui cherche la diversité en générant des solutions initiales aléatoires et en n'apprenant rien du passé. Dans la recherche avec tabous, la diversification permet de guider intelligemment l'exploration selon l'histoire de la recherche. Elle utilise des mémoires à long terme dans lesquelles elle enregistre certaines caractéristiques des solutions déjà rencontrées et en analysant ces caractéristiques, elle peut procéder à une modification du processus d'exploration afin de favoriser ou de pénaliser des caractéristiques communes et ainsi orienter la recherche. Si nous associons ces caractéristiques à la région déjà explorée, nous pouvons, en pénalisant ces caractéristiques, nous éloigner vers d'autres régions.

(Kelly, Laguna et Glover 1994), en explorant tout l'espace des solutions d'un problème d'optimisation combinatoire où il existe beaucoup de minima locaux. Les solutions visitées durant cette phase peuvent être non réalisables, mais nous pouvons utiliser des pénalités pour nous ramener dans le domaine réalisable.

Stratégie de chemins reliants

Cette stratégie, mentionnée dans la littérature sous l'appellation "path relinking" (Glover 1997, Glover, Laguna et Marti 2000 , Laguna et Marti 1999), présente une intégration intéressante des stratégies d'intensification et de diversification. Cette stratégie établit un sous-ensemble de solutions élites, à partir d'un ensemble de solutions, ainsi que des règles pour explorer les chemins entre les solutions de cet ensemble, dans le but de générer de nouvelles solutions. L'ensemble des solutions élites doit être défini pour assurer la qualité ou la diversité ou encore un mélange des deux caractéristiques. Par exemple, l'ensemble des solutions élites peut contenir les solutions qui minimisent le plus la valeur de la fonction objective. Pour générer de nouvelles solutions, des chemins sont explorés dans l'espace de voisinage entre une solution élite, choisie comme point de départ et chacune des autres solutions élites. Le rôle de ces solutions est interchangeable : chaque solution élite peut être considérée comme point de départ. Donc, à partir d'un point de départ, le but principal de la recherche est de trouver un chemin vers une solution élite, appelée dirigeante, en réalisant des mouvements qui introduisent dans la solution courante des caractéristiques de la solution dirigeante. Cette exploration permet à la recherche de réaliser des mouvements non attrayants pour la fonction objective mais qui sont essentiels pour atteindre des solutions avec de bonnes caractéristiques.

"Path relinking" est dérivée d'une approche appelée "scatter search". Cette dernière est organisée de façon à extraire des informations qui ne se trouvent pas séparément dans les solutions élites, tout en faisant appel à des méthodes heuristiques auxiliaires pour choisir les éléments à être combinés afin de générer de nouvelles solutions. La forme originale de "scatter search" est décrite dans la figure 2.4.

Le fondement de "scatter search" est basé sur une méthode appelée "surrogate constraint" utilisée pour résoudre les problèmes d'optimisation en nombres entiers.

1. Initialiser $\Gamma = \phi$
   Générer une solution de départ et l'ajouter à Γ
2. A partir de Γ, générer des solutions à l'aide d'un processus heuristique.
   Ajouter ces solutions à Γ
3. Extraire un sous-ensemble H des meilleures solutions de Γ (solutions élites)
4. Créer de nouvelles solutions à partir des combinaisons linéaires
   des solutions de H
5. Extraire une collection des meilleures solutions trouvées en 4
   et ajouter les à Γ
6. Arrêter si une itération limite est atteinte
   Sinon, aller 2.

FIG. 2.4 – Scatter search

constraint" utilisée pour résoudre les problèmes d'optimisation en nombres entiers. Le but de cette méthode consiste à générer de nouvelles contraintes à partir de la combinaison des contraintes existantes pour restreindre le domaine du problème relaxé et ainsi améliorer la qualité de la solution engendrée. "Scatter search" constitue la version primale de cette méthode où les contraintes sont remplacées par des solutions.

2.2 Le problème de synthèse de réseau multiproduits avec capacités

Le problème de synthèse de réseau multiproduits avec capacités consiste à transporter des produits entre différentes paires d'origines et de destinations d'un réseau donné. Chaque arc du réseau possède une capacité totale, un coût de transport associé à chaque produit et un coût fixe applicable dès qu'un flot d'un produit quelconque y circule. Pour formuler le problème, nous considérons un graphe orienté $G = (N, A)$ où N est l'ensemble des noeuds et A est l'ensemble des arcs. Nous dénotons par f_{ij} le coût fixe à payer pour utiliser l'arc (i, j) et par y_{ij} la variable de synthèse indiquant si l'arc (i, j) appartient $(y_{ij} = 1)$ ou non $(y_{ij} = 0)$ au réseau. Nous dénotons respectivement par x_{ij}^p et c_{ij}^p la quantité du flot et le coût par unité du flot du produit p traversant l'arc (i, j) et par P l'ensemble de produits à transporter. Notre modèle est alors le suivant (Magnanti et Wong 1984) :

$$\min z(x, y) = \sum_{(i,j)\in A} f_{ij} y_{ij} + \sum_{p\in P} \sum_{(i,j)\in A} c_{ij}^p x_{ij}^p \qquad (2.2)$$

Sujet à :

$$\sum_{j \in N^+(i)} x_{ij}^p - \sum_{j \in N^-(i)} x_{ji}^p = d_i^p \quad \forall i \in N, p \in P \tag{2.3}$$

$$\sum_{p \in P} x_{ij}^p \leq u_{ij} y_{ij} \quad \forall (i,j) \in A \tag{2.4}$$

$$x_{ij}^p \geq 0 \quad \forall (i,j) \in A, p \in P \tag{2.5}$$

$$y_{ij} = \{0,1\} \quad \forall (i,j) \in A \tag{2.6}$$

$$d_i^p = \begin{cases} -w_i^p & si\ i \text{ est une destination du produit } p \\ w_i^p & si\ i \text{ est une source du produit } p \\ 0 & sinon \end{cases}$$

u_{ij} : la capacité de l'arc (i,j),
w_i^p : l'offre du produit p au sommet i,
$-w_i^p$: la demande du produit p au sommet i,
$N^+(i)$: ensemble des successeurs du sommet i
$N^-(i)$: ensemble des prédécesseurs du sommet i.

Le nombre de sources et de destinations, ainsi que le nombre de produits est défini dépendemment des contextes. Dans un problème de réseau informatique, une source centrale alimente plusieurs terminaux (généralement on est en présence d'un graphe non orienté) ; dans le cas d'un problème de transport de marchandises, plusieurs sources (dépôts, gares) alimentent plusieurs destinations (clients, gares).

Pour adapter notre formulation au cas des graphes non orientés, nous remplaçons chaque occurence x_{ij}^p dans ces contraintes par $x_{ij}^p + x_{ji}^p$, limitant ainsi la quantité du flot circulant dans les deux directions à la capacité de l'arête (i,j).

Pour résoudre le problème de synthèse de réseau multiproduits avec capacités, notre objectif consiste à choisir les arcs à inclure dans le réseau pour répondre aux exigences de demandes et de capacités à moindre coût. Les contraintes (2.3) imposées pour chaque produit sont les contraintes de conservation du flot. Les contraintes (2.4) assurent que la quantité totale du flot de tous les produits sur l'arc (i,j) ne peut dépasser la capacité de l'arc s'il est ouvert et exigent un flot nul si l'arc (i,j) est fermé. Ces contraintes ont pour effet de forcer les variables continues du flot de

chaque produit à partager la capacité limitée des arcs empêchant ainsi de résoudre le problème en le séparant par produit. Les contraintes (2.4) lient également les variables continues du flot et les variables binaires de synthèse, nécessitant ainsi de considérer les deux types de variables simultanément lors de la résolution de ce problème.

2.2.1 Problématique

Malgré son importance socio-économique, le problème de synthèse de réseau multi-produits avec capacités demeure extrêmement difficile à résoudre. C'est un problème NP-complet. Pour qualifier cette complexité, il est utile de considérer des cas particuliers. Notre problème renferme comme cas spécial plusieurs problèmes qui sont NP-complets, en particulier : le problème de Steiner (Garey et Johnson 1979) et le problème de localisation sans capacité (Krarup et Pruzan 1983).

Outre la complexité théorique prouvée, la difficulté à résoudre notre problème revient en partie à sa nature fortement combinatoire. Les contraintes (2.4) compliquent grandement le problème pour deux raisons : d'une part, elles lient entre elles les variables continues (flot) et les variables binaires; d'autres part, elles lient également entre eux les produits et rendent ainsi le problème non décomposable par produit.

Une autre difficulté réside dans le conflit entre les coûts fixes et les coûts variables pour le choix d'une bonne solution, ainsi qu'entre les capacités limitées et les coûts fixes pour le choix des arcs à utiliser. Ouvrir plusieurs arcs dans le réseau fait augmenter les coûts fixes, mais le domaine des variables continues peut devenir plus grand, diminuant ainsi la valeur de la fonction objective associée. Par contre, avec peu d'arcs dans le réseau, les coûts fixes diminuent mais le domaine des variables continues se rétrécit, augmentant ainsi sa valeur objective. D'un autre côté, un coût fixe réduit pour un arc est attirant, mais si sa capacité n'est pas suffisante, sa chance d'être présent dans la solution optimale est minimale.

La relaxation continue de notre problème donne une solution loin de la solution optimale parce qu'elle ne considère pas le coût fixe d'une manière exacte. Pour obtenir la relaxation continue, nous remplaçons les contraintes (2.6) par :

$$0 \leq y_{ij} \leq 1 \quad \forall (i,j) \in A \tag{2.7}$$

Ces contraintes, combinées avec les contraintes (2.4), donnent :

$$\sum_{p \in P} x_{ij}^p / u_{ij} \leq y_{ij} \leq 1 \tag{2.8}$$

Puisque $f_{ij} \geq 0$, alors il existe une solution optimale avec $\sum_{p \in P} x_{ij}^p / u_{ij} = y_{ij}$. D'où la relaxation continue de notre problème peut s'écrire sous la forme :

$$\min \sum_{(i,j) \in A} \sum_{p \in P} (c_{ij}^p + f_{ij}/u_{ij}) x_{ij}^p \tag{2.9}$$

Sujet à :

$$\sum_{j \in N^+(i)} x_{ij}^p - \sum_{j \in N^-(i)} x_{ji}^p = d_i^p \quad \forall i \in N, p \in P \tag{2.10}$$

$$\sum_{p \in P} x_{ij}^p \leq u_{ij} \quad \forall (i,j) \in A \tag{2.11}$$

$$x_{ij}^p \geq 0 \quad \forall (i,j) \in A, p \in P \tag{2.12}$$

En pratique, les arcs ne sont pas tous saturés dans la solution optimale alors seulement la fraction $\sum_{p \in P} x_{ij}^p / u_{ij}$ du coût fixe est incluse dans la fonction objective. Etant donné la mauvaise qualité de la borne générée, les méthodes de résolution basées sur des relaxations linéaires ne sont plus intéressantes lorsqu'appliquées directement sur la formulation.

Pour un réseau composé de n noeuds, m arcs et p produits, notre formulation compte : $np+m$ contraintes et $mp+m$ variables. Pour un problème qui comprend $n = 50$, $m = 700$ et $p = 400$, nous obtenons 20700 contraintes et 280700 variables, ce qui est trop grand pour être résolu par une méthode exacte d'optimisation combinatoire.

Toutes ces difficultés favorisent l'utilisation des méthodes heuristiques robustes et flexibles qui ont prouvé leur efficacité à obtenir de bonnes solutions pour des problèmes complexes.

2.2.2 Problèmes apparentés

Plusieurs problèmes apparentés au problème de synthèse de réseau multiproduits avec capacités existent dans la littérature. Certains d'entres eux présentent des cas

particuliers de notre formulation, alors que d'autres sont obtenus soit en ajoutant des contraintes additionnelles ou encore en intrégrant à la fonction objective des coûts non linéaires. Parmi les cas particuliers, on retrouve : l'arbre sous-tendant minimal, le problème du flot à coût minimum, le problème de Steiner, le problème de voyageur de commerce (Magnanti et Wong 1984).

Pour obtenir le problème du flot à coût minimum, nous posons $D = \phi$ et $P = \{1\}$. Plusieurs algorithmes existent pour la résolution de ce problème (Ahuja, Magnanti, et Orlin 1993). En particulier, un algorithme primal construit à partir d'une solution réalisable x, le graphe résiduel associé $\bar{G}(x)$, puis améliore la valeur de la fonction objective à chaque itération en augmentant le flot à travers les circuits négatifs (Ahuja, Magnanti, et Orlin 1993). La convergence de cet algorithme découle du théorème suivant : une condition nécessaire et suffisante pour qu'un flot x soit optimal est qu'il n'existe aucun circuit négatif dans le graphe résiduel associé $\bar{G}(x)$.

Le graphe résiduel $\bar{G}(x)$ est déterminé à partir de G en considérant le même ensemble de sommets N et en remplaçant chaque arc $(i,j) \in A$ par deux arcs $(i,j)^+$ et $(j,i)^-$ dans $\bar{G}$. Les coûts et les capacités résiduelles de ces arcs sont calculés de la façon suivante :

$$\begin{array}{llll} c_{ij}^{+} = c_{ij} & r_{ij} = u_{ij} - x_{ij} & si & x_{ij} < u_{ij} \\ c_{ji}^{-} = -c_{ij} & r_{ji} = x_{ij} & si & x_{ij} > 0 \end{array}$$

Pour détecter un circuit négatif, nous pouvons utiliser l'algorithme d'ajustement des étiquettes "label correcting" (Ahuja, Magnanti et Orlin 1993). L'algorithme primal qui combine les ingrédients du problème de plus court chemin, ainsi que celles du problème du flot à valeur maximale sera utilisé pour la résolution de notre problème (voir chapitre 3).

Le problème de synthèse de réseau sans capacité est aussi un cas particulier de notre formulation. Pour obtenir ce modèle, il suffit de borner inférieurement la capacité de chaque arc par la somme de la demande de tous les produits ($u_{ij} \geq 1/2 \sum_{p \in P} \sum_{i \in N} |d_i^p|$, $\forall (i,j) \in A$). Dans ce cas, les contraintes de capacité deviennent redondantes lorsque $y_{ij} = 1$. Le problème de synthèse de réseau sans capacité a

été étudié par plusieurs chercheurs. En particulier, Balakrishnan, Magnanti et Wong (1989) ont travaillé sur la version forte de notre modèle (dans le cas d'un graphe non orienté) où ils remplacent les contraintes de capacité par les contraintes :

$$x_{ij}^p \leq y_{ij} \quad \forall p \in P, (i,j) \in A. \tag{2.13}$$

Dans leur article, ils présentent une procédure duale ascendante qui, en conjonction avec une heuristique "add-drop", est capable de résoudre efficacement des problèmes de grande taille. Les résultats obtenus sont impressionnants : l'écart entre les bornes générées est d'au plus 4% sur des problèmes ayant jusqu'à 500 variables binaires et près de 2 millions de variables continues. Malheureusement, les chercheurs concluent que la généralisation de leur méthode n'est pas effective pour résoudre des problèmes avec capacités. Ils notent aussi que la relaxation continue ne donne plus une bonne approximation du problème entier.

Holmberg et Hellstrand (1998) proposent une heuristique lagrangienne dans le contexte d'une méthode de séparation et évaluation progressive ("branch-and-bound") pour résoudre le même problème. L'heuristique lagrangienne utilise une relaxation lagrangienne comme sous problème, résout cette relaxation par la méthode de sous-gradient puis génère des solutions réalisables par une méthode heuristique primale. Lorsque la marge duale entre les meilleures bornes inférieures et supérieures est suffisement petite, les auteurs utilisent la méthode "branch-and-bound" pour trouver des solutions optimales. Les résultats numériques ont démontré l'efficacité de la méthode pour résoudre les problèmes de grande taille avec plusieurs produits.

Hellstrand, Larsson et Migdalas (1992) ont réalisé une étude polyédrale et démontrent que le polytope obtenu de la relaxation linéaire de la fomulation forte du problème possède la propriété de quasi-intégralité, dans le sens que chaque arête de l'enveloppe convexe des points entiers est aussi une arête du polytope lui-même. Cette propriété induit la possibilité de résoudre ce problème par une méthode de pivotage.

Un autre cas particulier est le problème de localisation de dépôts, qui consiste à déterminer parmi un ensemble de sites potentiels de dépôts, un sous-ensemble qui permet de satisfaire les demandes des clients en respectant les capacités des sites de

dépôts choisis. L'objectif consiste à minimiser le coût total composé du coût d'installation des dépôts (i.e, les coûts fixes) et des coûts de transport. Dans ce problème, l'ensemble des sommets est partitionné en deux ensembles qui présentent les sites potentiels de dépôts et les clients à servir. Pour convertir ce problème en un problème de synthèse de réseau, nous remplaçons chaque sommet j, candidat à être un dépôt, par deux sommets, un sommet récepteur j_1, et un autre donneur j_2. Nous ajoutons l'arc (j_1, j_2) au réseau et nous lui associons un coût fixe égal au coût de construction du dépôt et un coût de transport nul. Nous ajoutons une capacité si le dépôt j a une capacité limitée. Nous remplaçons chaque arc (i, j) par un arc (i, j_1) et chaque arc (j, k) par un arc (j_2, k). Dans ce cas, un dépôt j potentiel est choisi si la variable binaire $y_{j_1 j_2}$ est mise à 1.

Cornuéjols, Sridharan et Thizy (1991) étudient plusieurs relaxations du problème de localisation avec capacité et comparent la qualité des différentes bornes inférieures obtenues. De plus, ils comparent les solutions réalisables obtenues par plusieurs de ces relaxations aux solutions obtenues par des algorithmes voraces ou des heuristiques d'échanges. Le problème de localisation sans capacité est un cas spécial de ce modèle, étudié notamment dans l'article de Krarup et Pruzan (1983).

Le problème de localisation avec équilibrage est aussi un cas particulier du problème de synthèse de réseau multiproduits avec capacités. Celui-ci a été formulé pour la première fois par Crainic, Dejax et Delorme (1989) et consiste à localiser les dépôts de conteneurs à utiliser, à établir l'affectation des clients aux dépôts pour chaque type de conteneurs, et à planifier les flots de conteneurs vides entre les dépôts de façon à équilibrer les différences entre l'offre et la demande de conteneurs dans les diverses régions du territoire desservi par l'entreprise de transport par conteneurs. Des travaux ont permis de résoudre efficacement le problème lorsque les dépôts ont une capacité illimitée (Gendron et Crainic 1995, 1997, Gendron, Potvin, et Soriano 1999, Crainic et Delorme 1993). En particulier, Crainic et al. (1993) adaptent une méthode de recherche avec tabous pour la résolution de ce problème et montrent que leur approche trouve d'excellentes solutions dans un temps raisonnable, proposant ainsi de l'étendre pour d'autres problèmes de programmation mixte. Pour définir leur structure de voisinage, Crainic et al. (1993) changent, à partir de la solution courante, la valeur d'une

variable de synthèse y_j (associée à un dépôt). Dans leur approche, ouvrir un dépôt peut affecter la distribution du flot d'une façon considérable, puisqu'en général peu de dépôts existent dans une solution. Ceci n'est pas le cas dans notre modèle où un nombre considérable d'arcs est présent. Par suite, ajouter ou éliminer un arc peut n'avoir aucun impact sur la distribution du flot dans le réseau.

Gendron, Potvin et Soriano (2003) proposent de combiner une méthode de recherche avec tabous avec une méthode de programmation mathématique pour résoudre ce problème dans le cas où les dépôts ont une capacité limitée. Les auteurs utilisent la même structure de voisinage développée par Crainic et al. (1993) et montrent que leur approche engendre de bonnes solutions dans des temps de calculs compétitifs lorsqu'on la compare aux résultats obtenus par un logiciel commercial de programmation en nombres entiers.

Lorsqu'on veut considérer des fréquences sur les arcs, les variables de synthèse ne sont plus restreintes à des valeurs binaires mais sont remplacées par les contraintes suivantes :

$$y_{ij} \geq 0 \text{ et entière.} \tag{2.14}$$

Ainsi, une valeur $y_{ij} > 1$ indique le nombre de services (trains, autobus, camion,...) offerts sur cette ligne (l'arc (i, j)). Les modèles étudiant cette notion de fréquence sont nombreux. Le problème de transport ferroviaire des marchandises est un exemple. Des travaux ont été effectués pour modéliser et résoudre ce problème (Crainic et Rousseau 1986, Crainic, Ferland et Rousseau 1984). Crainic et Rousseau (1986) présentent un algorithme heuristique qui détermine d'une façon itérative les fréquences des services de train à offrir, puis compte tenu des fréquences des services offertes, établit une distribution optimale du flot jusqu'à ce que la valeur de la fonction objective soit inférieure à un seuil fixé. Cependant, dans leur modèle, les contraintes de capacités ne sont pas traitées d'une manière explicite mais sont plutôt représentés par un terme de pénalité intégré à l'objectif.

D'autres modèles de synthèse de réseau permettent d'aller au delà de la capacité réelle d'un arc en installant différents types de capacités. Ces modèles apparaissent dans plusieurs applications du domaine du transport (Leung, Magnanti et Singhal 1990), de télécommunications et dans d'autres industries. Le problème consiste à uti-

liser la capacité réelle d'un réseau, et si nécessaire, le renforcer en installant différents types de capacités pour satisfaire la demande à moindre coût. Berger et al. (2000) développent une méthode de recherche avec tabous et utilisent la formulation de chemins pour résoudre ce problème. Dans leur modèle, une source centrale alimente le réseau et un seul chemin est permis pour satisfaire chaque demande. Pour définir la structure de voisinage, les chercheurs génèrent pour chaque chemin appartenant à la solution courante, k plus court chemins pour passer à la solution voisine. L'utilisation d'une méthode de recherche avec tabous permet aux chercheurs de considérer plusieurs types de capacités et les résultats obtenus sont meilleurs que ceux obtenus par les heuristiques d'échanges qui s'arrêtent au premier minimum local rencontré. Un aspect important de la recherche avec tabous employé dans cet article est l'utilisation d'une mémoire adaptative qui enregistre les meilleurs chemins utilisés de la source aux sommets demandes et de les combiner pour relancer la recherche.

Cependant, la majorité des méthodes de résolution pour les problèmes d'installations sur les arcs est basée sur le principe de renforcer la relaxation linéaire du problème en ajoutant des inégalités valides et des facettes (Magnanti, Mirchandani et Vachani 1993, Magnanti et Mirchandani 1993, Chopra, Gilboa et Sastry 1998, Bienstock et Günlük 1996, Barahona 1996, Epstein 1998). En particulier, Magnanti, Mirchandani et Vachani (1995) considèrent seulement deux types de capacités et un coût de transport nul pour chaque produit. Ils génèrent trois classes d'inégalités valides et démontrent que la formulation linéaire incluant ces inégalités donnent une approximation de la borne inférieure obtenue par la relaxation lagrangienne en relaxant les contraintes de conservation de flot. Cependant, les chercheurs signalent que la marge duale est encore grande et qu'il est nécessaire d'identifier de nouvelles inégalités valides pour améliorer la performance des méthodes de coupes.

Dans certains modèles de synthèse de réseau, la fonction objective est non linéaire et convexe pour tenir compte d'effets de congestion ou de délai présent dans le réseau. Dans d'autres, cette fonction est concave. Pour une revue détaillée des problèmes de synthèse de réseau avec fonction non linéaire, voir Magnanti et Wong (1984) et Guisewite et Pardalos (1990). Dans les modèles présentant une fonction concave avec des contraintes linéaires, une solution optimale se trouve à un point extrême du do-

maine réalisable. Gallo et Sodini (1979) ont essayé de tirer profit de cette propriété et ont appliqué des heuristiques pour résoudre ces problèmes. Leur voisinage est défini par les points extrêmes qui sont adjacents à la solution courante. Ainsi, à partir d'une solution initiale, l'algorithme se déplace vers la meilleure solution adjacente (point extrême) jusqu'à ce qu'un minimum soit atteint. Bazlamaçci et Hindi (1996), ont amélioré la structure de voisinage développée par Gallo et Sodini (1979) et ont développé une méthode de recherche avec tabous pour éviter le piège des minima locaux. Les résultats numériques ont indiqué la supériorité de cette approche par rapport à la première.

2.2.3 Revues des méthodes de résolution

Dans la littérature, nous distinguons trois approches principales pour résoudre le problème de synthèse de réseau multiproduits avec capacités : les méthodes de coupes, les méthodes duales et les heuristiques. Dans ce qui suit, nous décrivons chacune de ces méthodes. Nous décrivons également les points forts et les points faibles de chacune.

Les méthodes de coupes

Les méthodes de coupes commencent à résoudre la relaxation continue en relaxant les contraintes d'intégralité, puis essayent d'améliorer la formulation en ajoutant des inégalités valides. Ces inégalités représentent des coupes qui séparent la solution obtenue du problème relaxé du domaine réalisable du problème original. L'efficacité d'une telle approche réside dans le fait qu'on dispose d'un algorithme de résolution robuste, la méthode du simplexe, pour résoudre les problèmes relaxés et qu'on améliore les bornes générées moyennant l'ajout des inégalités valides. Par contre, cette approche n'exploite aucune structure du problème ce qui la rend très sensible à sa taille.

Chouman, Crainic et Gendron (2003a, 2003b) ont étudié cette approche dans le but d'améliorer la qualité de la borne inférieure obtenue de la relaxation linéaire du problème de synthèse de réseau multiproduits avec capacités. Les chercheurs présentent, dans un premier temps une revue de trois familles d'inégalités valides proposées dans la littérature qui sont pertinentes pour le problème puis, développent un algorithme de coupes qui intègre ces inégalités. Les résultats numériques ont in-

diqué que l'agorithme proposé génère de meilleures bornes inférieures dans des temps compétitifs lorsqu'on le compare à CPLEX.

Les méthodes duales

Les méthodes duales, comme les relaxations lagrangiennes, permettent d'obtenir des bornes inférieures sur l'objectif en résolvant des sous-problèmes plus simples que le modèle original. Ces sous-problèmes sont obtenus en relaxant certaines des contraintes et en les introduisant dans l'objectif avec des facteurs de pénalités (multiplicateurs). Les méthodes duales exploitent la structure du problème mais posent un problème à optimiser le dual lagrangien. Par exemple, les méthodes de sous-gradient ne montrent pas une convergence continue, elles montrent du "zigzag" et parfois elles s'arrêtent loin de la solution optimale. Pour vaincre ce problème, les chercheurs ont mis au point d'autres méthodes, notemment les méthodes des faisceaux (LeMaréchal 1989). Celles-ci interdisent le "zigzag" en assurant toujours une direction de montée mais elles sont plus difficiles à implémenter. Pour améliorer la borne obtenue par la relaxation lagrangienne, les chercheurs ajoutent des inégalités valides mais ces dernières posent parfois l'inconvénient de détruire la structure du problème.

Plusieurs chercheurs ont essayé de résoudre notre problème par des relaxations lagrangiennes et du "branch-and-bound" : Homberg et Yuan (2000), Gendron, Crainic et Frangioni (1998), Crainic, Frangioni et Gendron (2001), Sellmann, Kliewer et Koberstein (2002). Plus particulièrement, Gendron et Crainic (1994) étudient trois formulations différentes du problème et comparent les bornes inférieures obtenues de trois formulations en utilisant trois relaxations différentes : la première relaxation est la relaxation continue (en relaxant les contraintes d'intégralité). La deuxième est la relaxation lagrangienne dans laquelle ils considèrent deux cas : le premier en relaxant les contraintes liantes puis le deuxième en relaxant les contraintes de conservation de flot. Finalement, la troisième relaxation qu'ils considèrent pour les trois formulations est la décomposition lagrangienne. Gendron et Crainic (1994) concluent que les meilleurs résultats sont obtenus en relaxant les contraintes liantes. Cependant, la marge duale obtenue est encore très grande particulièrement pour les problèmes qui présentent un coût fixe élevé.

Gendron, Crainic et Frangioni (1998) présentent un modèle général du problème de synthèse de réseau multiproduits avec capacités. Ils donnent une revue de la littérature sur les méthodes de résolution de ce problème et suggèrent qu'une procédure de résolution efficace pour ce problème difficile consiste à combiner les méthodes de coupes et de relaxation lagrangienne avec des heuristiques sophistiquées comme la recherche avec tabous.

Les heuristiques

Les heuristiques sont certainement requises pour résoudre les problèmes complexes d'optimisation combinatoire. Elles ont prouvé à maintes reprises leur efficacité dans ce domaine, mais elles dépendent essentiellement de la modélisation du problème. Par exemple, dans le contexte d'une méthode de recherche locale, il faut définir une bonne structure de voisinage, une méthode efficace et économique pour évaluer les voisins de la solution courante et elles nécessitent la calibration de plusieurs paramètres.

Crainic, Gendreau et Farvolden (2000) ont développé une méthode de recherche avec tabous pour trouver des bornes supérieures de qualité pour notre problème. Ils ont utilisé la formulation de chemin suivante :

$$\min z(h,y) = \sum_{a \in D} f_a y_a + \sum_{p \in P} \sum_{l \in L^p} k_l^p h_l^p \tag{2.15}$$

Sujet à :

$$\sum_{l \in L^p} h_l^p = w^p \quad \forall p \in P \tag{2.16}$$

$$\sum_{p \in P} \sum_{l \in L^p} h_l^p \delta_{al}^p \leq u_a y_a \quad \forall a \in D \tag{2.17}$$

$$\sum_{p \in P} \sum_{l \in L^p} h_l^p \delta_{al}^p \leq u_a \quad \forall (i,j) \in A \setminus D \tag{2.18}$$

$$h_l^p \geq 0 \quad \forall l \in L^p, p \in P \tag{2.19}$$

$$y_a = \{0,1\} \quad \forall a \in D \tag{2.20}$$

Avec

L^p = ensemble des chemins du produit $p \in P$,
h_l^p = flot du produit $p \in P$ sur le chemin $l \in L^p$,
δ_{al}^p = 1 si l'arc a appartient au l^{iem} chemin du produit p et 0 sinon,
k_l^p = $\sum_{a \in A} c_a^p \delta_{al}^p$ représente le coût du chemin $l \in L^p$ du produit $p \in P$.

Leur approche consiste à explorer l'espace des variables continues de flot, et par suite, la valeur d'une variable de synthèse y_{ij} est mise à 1 s'il existe un flot qui traverse l'arc (i, j) et à 0 sinon. Crainic, Gendreau et Farvolden (2000) proposent une procédure itérative en deux phases : une phase locale et une phase de diversification. Les résultats numériques obtenus sont intéressants; cependant, l'approche proposée est limitée par le fait que chaque mouvement n'affecte que la déviation du flot d'un seul produit et qu'elle ne garantit pas que tous les chemins soient considérés ni implicitement ni explicitement.

Crainic, Gendron et Hernu (2002) proposent une heuristique d'ajustement de pente pour résoudre notre problème. L'heuristique commence par générer rapidement des solutions réalisables à partir d'une méthode itérative qui résout une approximation linéaire de la formulation du problème. Pour améliorer la relative mauvaise qualité des solutions produites, l'heuristique combine une perturbation lagrangienne et des procédures d'apprentissage basées sur des structures de mémoires à long terme. Les résultats numériques obtenus montrent la nécessité d'utiliser des mémoires à long terme pour améliorer l'efficacité globale de l'algorithme.

Homberg et Yuan (2000) proposent de résoudre notre problème par une heuristique lagrangienne utilisée dans une méthode d'énumération de type "branch-and-bound". L'heuristique lagrangienne utilisée est composée d'une relaxation lagrangienne (par rapport aux contraintes de conservation de flot), une procédure efficace de sous-gradient pour résoudre le dual lagrangien et d'une heuristique primale pour trouver des solutions réalisables. Puisque, pour les problèmes de grande taille, il est impossible en temps raisonnable, d'appliquer exactement la méthode "branch-and-bound", l'idée des chercheurs était de guider le branchement pour investir seulement aux noeuds les plus prometteurs. Ceci est réalisé en fixant un certain nombre de variables de synthèse avant chaque branchement. L'heuristique lagrangienne est utilisée pour identifier des bornes inférieures et des solutions réalisables à chaque noeud. En plus, les solutions correspondantes aux bornes inférieures générées par la suite de

sous-gradients sont utilisées, entre autres, pour déterminer la liste des variables de synthèse à être fixées. Deux heuristiques sont utilisées pour ce faire. La première, appelée α-fixing, considère le nombre de fois que l'arc (i,j) a eu le même statut dans les solutions des sous-problèmes lagrangiens. Si ce nombre dépasse une certaine limite, on fixe le statut de l'arc (i,j). Ainsi, étant donné $\alpha \in [0, 0.5]$ et M le nombre de sous-problèmes lagrangiens, alors :

$$\begin{array}{lll} y_{ij} & \text{est fixé à } 1 \text{ si} & \sum_{l=1,M} y_{ij}^l \geq (1-\alpha)M \\ y_{ij} & \text{est fixé à } 0 \text{ si} & \sum_{l=1,M} y_{ij}^l \leq \alpha M \end{array}$$

où y_{ij}^l dénote la valeur de y_{ij} dans la solution l du sous-problème lagrangien.

La deuxième heuristique, appelée β-fixing, avec $\beta \in [0,1]$, considère le coût réduit $\hat{g}_{ij}$ de l'arc plutôt que son statut. Dans cette heuristique, si n dénote le nombre initial d'arcs, alors exactement $[\beta n]$ arcs seront fixés à chaque branchement. Si ce nombre dépasse le nombre d'arcs non encore fixés, tous les arcs seront alors fixés. Les coûts réduits $\hat{g}_{ij}$ seront cumulés durant les itérations de sous-gradients d'une façon spéciale puis triés par ordre décroissant selon leur valeur absolue. Les $[\beta n]$ premiers arcs auront leur statut fixé à ouvert si leur coût cumulé est strictement négatif ou à fermer si leur coût cumulé est positif. Pour cumuler les coûts réduits, l'heuristique commence par initialiser R_{ij} à $\hat{g}_{ij}^1$ à la première itération ; ensuite, quand la meilleure borne inférieure est améliorée, disons à l'itération l, le coût cumulé devient $R_{ij} = \gamma R_{ij} + \hat{g}_{ij}^l$ avec $\gamma \in [0,1]$, γ qui sert à donner moins de poids à l'ancien coût réduit (les chercheurs ont fixé la valeur de γ à 0.5).

Les heuristiques α-fixing et β-fixing utilisent les informations générées par la relaxation. Cependant, avec le β-fixing, la taille de l'arbre du "branch-and-bound" est connue à l'avance. Par exemple, si $\beta = 0.1$, alors 10% des variables de synthèse sont fixées à chaque niveau, ce qui donne un arbre de profondeur égale à 10. Il est donc plus facile de contrôler le temps d'exécution avec cette approche, plutôt qu'avec la première.

2.3 Conclusion

Nous avons présenté dans ce chapitre la méthode de recherche avec tabous. Nous avons également présenté un survol des travaux reliés au problème de synthèse de réseau multiproduits avec capacités. Nous avons discuté des principales approches de résolution rapportées dans la littérature. Nous pouvons conclure que le problème de synthèse de réseau multiproduits avec capacités pose d'importants défis de recherche, et que pour être en mesure de trouver des solutions de bonne qualité pour des instances de grande taille, il s'avère nécessaire de développer des métaheuristiques, comme la méthode de recherche avec tabous.

Bibliographie

Ahuja, R.K., Magnanti, T.L., and Orlin, J.B. (1993). *Network Flows – Theory, Algorithms, and Applications.* Prentice-Hall, Englewood Cliffs, NJ.

Balakrishnan, A., Magnanti, T.L., and Wong, R.T. (1989). A Dual–Ascent Procedure for Large–Scale Uncapacitated Network Design. *Operations Research*, 37(5) :716–740.

Barahona F. (1996). Network Design Using Cut Inequalities. *SIAM Journal of Optimization*, 6 :823–837.

Bazlamaçci C. and Hindi K. S. (1996). Enhanced Adjacent Extreme Point Search and Tabu Search for the Minimum, Concave-Cost Uncapacitated Transshipment Problem. *Journal of the Operational Research Society*, 47 :1150–1165.

Berger, D., Gendron, B., Potvin, J.-Y., Raghavan, S., and Soriano, P. (2000). Tabu Search for a Network Loading Problem with Multiple Facilities. *Journal of Heuristics*, 6(2) :253–267.

Bienstock D. and Günlük O. (1996). Capacitated Network Design-Polyedral Structure and Computation. *INFORMS Journal on Computing*, 8 :243–259.

Chopra S., Gilboa I., and Sastry S.T. (1998). Source Sink Flows with Capacity Installation in Batches. *Discrete Applied Mathematics*, 85 :165–192.

Chouman, M., Crainic, T.G., and Gendron, B. (2003a). A Cutting-Plane Algorithm Based on Cutset Inequalities for Multicommodity Capacitated Fixed Charge Network Design . Technical report, Centre de recherche sur les transports, Université de Montréal, Montréal, QC, Canada.

Chouman, M., Crainic, T.G., and Gendron, B. (2003b). Revue des inégalités valides pertinentes aux problèmes de conception de réseaux. *INFOR*, 41 :5–33.

Bibliographie

Ahuja, R.K., Magnanti, T.L., and Orlin, J.B. (1993). *Network Flows – Theory, Algorithms, and Applications.* Prentice-Hall, Englewood Cliffs, NJ.

Balakrishnan, A., Magnanti, T.L., and Wong, R.T. (1989). A Dual–Ascent Procedure for Large–Scale Uncapacitated Network Design. *Operations Research*, 37(5) :716–740.

Barahona F. (1996). Network Design Using Cut Inequalities. *SIAM Journal of Optimization*, 6 :823–837.

Bazlamaçci C. and Hindi K. S. (1996). Enhanced Adjacent Extreme Point Search and Tabu Search for the Minimum, Concave-Cost Uncapacitated Transshipment Problem. *Journal of the Operational Research Society*, 47 :1150–1165.

Berger, D., Gendron, B., Potvin, J.Y., Raghavan, S., and Soriano, P. (2000). Tabu Search for a Network Loading Problem with Multiple Facilities. *Journal of Heuristics*, 6(2) :253–267.

Bienstock D. and Günlük O. (1996). Capacitated Network Design-Polyedral Structure and Computation. *INFORMS Journal on Computing*, 8 :243–259.

Chopra S., Gilboa I., and Sastry S.T. (1998). Source Sink Flows with Capacity Installation in Batches. *Discrete Applied Mathematics*, 85 :165–192.

Chouman, M., Crainic, T.G., and Gendron, B. (2003a). A Cutting-Plane Algorithm Based on Cutset Inequalities for Multicommodity Capacitated Fixed Charge Network Design . Technical report, Centre de recherche sur les transports, Université de Montréal, Montréal, QC, Canada.

Chouman, M., Crainic, T.G., and Gendron, B. (2003b). Revue des inégalités valides pertinentes aux problèmes de conception de réseaux. *INFOR*, 41 :5–33.

Cornuéjols, G., Sridharan, R., and Thizy, J.M. (1991). A Comparison of Heuristics and Relaxations for the Capacitated Plant Location Problem. *European Journal of Operational Research*, 50 :280–297.

Crainic, T.G., Dejax, P.J., and Delorme, L. (1989). Models for Multimode Multicommodity Location Problems with Interdepot Balancing Requirements. *Annals of Operations Research*, 18 :279–302.

Crainic, T.G. and Delorme, L. (1993). Dual-Ascent Procedures for Multicommodity Location-Allocation Problems with Balancing Requirements. *Transportation Science*, 27(2) :90–101.

Crainic, T.G., Ferland, J.-A., and Rousseau, J.-M. (1984). A Tactical Planning Model for Rail Freight Transportation. *Transportation Science*, 18(2) :165–184.

Crainic, T.G., Frangioni, A., and Gendron, B. (2001). Bundle-Based Relaxation Methods for Multicommodity Capacitated Network Design. *Discrete Applied Mathematics*, 112 :73–99.

Crainic, T.G., Gendreau, M., and Farvolden, J.M. (2000). A Simplex-Based Tabu Search Method for Capacitated Network Design. *INFORMS Journal on Computing*, 12(3) :223–236.

Crainic, T.G., Gendreau, M., Soriano, P., and Toulouse, M. (1993). A Tabu Search Procedure for Multicommodity Location/Allocation with Balancing Requirements. *Annals of Operations Research*, 41 :359–383.

Crainic, T.G, Gendron, B, and Hernu, G. (2002). A slope Scaling/Lagrangean Perturbation Heuristic with Long-Term Memory for Multicommodity Capacitated Fixed-Charge Network Design. Publication, Centre de recherche sur les transports, Université de Montréal, Montréal, QC, Canada.

Crainic, T.G. and Rousseau, J.-M. (1986). Multicommodity, Multimode Freight Transportation : A General Modeling and Algorithmic Framework for the Service Network Design Problem. *Transportation Research B : Methodology*, 20B :225–242.

Epstein, R. (1998). *Linear Programming and Capacitated Network Loading.* PhD thesis, Sloan School of Management, Massachusetts Institute of Technology.

Gallo G. and Sodini C. (1979). Adjacent Extreme Flows and Application to Min Concave Cost Flow Problem. *Networks*, 9 :95–221.

Garey, M. and Johnson, D. (1979). *Computers and Intractability : A Guide to the Theory of NP-Completeness.* Freeman, San Francisco, CA.

Gendron, B., Crainic, T.G., and Frangioni, A. (1998). Multicommodity Capacitated Network Design. In Sansó, B. and Soriano, P., editors, *Telecommunications Network Planning*, pages 1–19. Kluwer Academic Publishers, Norwell, MA.

Gendron, B. and Crainic, T.G. (1994). Relaxations for Multicommodity Network Design Problems. Publication CRT-965, Centre de recherche sur les transports, Université de Montréal, Montréal, QC, Canada.

Gendron B. and Crainic T.G. (1995). A Branch-and-Bound Algorithm for Depot Location and Container Fleet Management. *Location Science*, 3 :39–53.

Gendron, B. and Crainic, T.G. (1997). A Parallel Branch-and-Bound Algorithm for Multicommodity Location with Balancing Requirements. *Computers & Operations Research*, 24(9) :829–847.

Gendron B., Potvin J.Y., and Soriano P. (1999). Tabu Search with Exact Neighbor Evaluation for Multicommodity Location with Balancing Requirements. *INFOR*, 37 :255–270.

Gendron B., Potvin J.Y., and Soriano P. (2003). A Tabu Search with Slope Scaling for the Multicommodity Capacitated Location Problem with Balancing Requirements. *Annals of Operations Research*, 122 :193–217.

Glover, F., Laguna, M., and Marti, R. (2000). Fundamentals of scatter search and path relinking. *Control and Cybernetics*, 39(3) :653–684.

Glover, F. and Laguna, M. (1997). *Tabu Search.* Kluwer Academic Publishers, Norwell, MA.

Glover, F. (1986). Future Paths for Integer Programming and Links to Artificial Intelligence. *Computers & Operations Research*, 1(3) :533–549.

Glover, F. (1989). Tabu Search – Part I. *ORSA Journal on Computing*, 1(3) :190–206.

Glover, F. (1990). Tabu Search – Part II. *ORSA Journal on Computing*, 2(1) :4–32.

Glover, F. (1996). Tabu Search and Adaptive Memory Programming – Advances, Applications and Challenges. In Barr, R., Helgason, R., and Kennington, J., editors, *Interfaces in Computer Science and Operations Research*, pages 1–75. Kluwer Academic Publishers, Norwell, MA.

Glover, F. (1997). A Template for Scatter Search and Path Relinking. In Hao, J., Lutton, E., Ronald, E., Schoenauer, M., and Snyers, D., editors, *Artificial Evolution*, volume 1363 of *Lecture Notes in Computer Science*, pages 13–54. Springer Verlag, Berlin.

Guisewite, G.M. and Pardalos, P.M. (1990). Minimum concave cost network flow problems : applications, complexity and algorithms. *Annals of Operations Research*, 25 :75–100.

Hellstrand J., Larsson T., and Migdalas A. (1992). A Characterization of the Uncapacitated Network Design Polytope. *Operations Research Letters*, 12 :159–163.

Holmberg, K. and Hellstrand, J. (1998). Solving the Uncapacitated Network Design Problem by a Lagrangian Heuristic and Branch-and-Bound. *Operations Research*, 46(2) :247–259.

Holmberg, K. and Yuan, D. (2000). A Lagrangean Heuristic Based Branch-and-Bound Approach for the Capacitated Network Design Problem. *Operations Research*, 48(3) :461–481.

Kelly, J.P., Laguna, M., and Glover, F. (1994). A Study of Diversification Strategies for the Quadratic Assignment Problem. *Computers & Operations Research*, 21(8) :885–893.

Krarup J. and Pruzan P.M. (1983). The Simple Plant Location Problem : Survey and Synthesis. *European Journal of Operational Research*, 12 :36–81.

Laguna, M. and Marti, R. (1999). GRASP and Path Relinking for 2-Layer Straight Line Crossing Minimization. *INFORMS Journal on Computing*, 11(1) :44–52.

Lemaréchal, C. (1989). Nondifferentiable Optimization. In Nemhauser, G.L., Rinnoy Kan, A.H.G., and Todd, M.J., editors, *Optimization*, volume 1 of *Handbooks in Operations Research and Management Science*, pages 529–572. North-Holland, Amsterdam.

Leung, J.M.Y., Magnanti, T.L., and Singhal, V. (1990). Routing in Point-to-Point Delivery Systems : Formulations and Solutions Heuristics. *Transportation Science*, 24(4) :245–260.

Magnanti, T.L., Mirchandani, P., and Vachani, R. (1993). The Convex Hull of Two Core Capacitated Network Design Problems. *Mathematical Programming*, 60 :233–250.

Magnanti, T.L., Mirchandani, P., and Vachani, R. (1995). Modeling and Solving the Two-Facility Capacitated Network Loading Problem. *Operations Research*, 43 :142–157.

Magnanti, T.L. and Mirchandani, P. (1993). Shortest Paths, Single Origine-Destination Network Desgin, and Associated Polyhedra. *Networks*, 23 :103–121.

Magnanti, T.L. and Wong, R.T. (1984). Network Design and Transportation Planning : Models and Algorithms. *Transportation Science*, 18(1) :1–55.

Sellmann, M., Kliewer, G., and Koberstein, A. (2002). Capacitated Network Design, Cardinality Cuts and Coupled Variable Fixing Algorithms based on Lagrangian Relaxations. Publication tr-ri-02-234, University of Paderborn, Department of Mathematics and Computer Science.

Chapitre 3

Cycle-Based Neighbourhoods for Fixed-Charge Capacitated Multicommodity Network Design

La référence de cet article est :
Ghamlouche I., Crainic T.G. and Gendreau M. "Cycle-Based Neighbourhoods for Fixed-Charge Capacitated Multicommodity Network Design", *Operations Research* 51(4), 2003.

Cycle-Based Neighbourhoods for Fixed-Charge Capacitated Multicommodity Network Design

Abstract

We propose new cycle-based neighbourhood structures for meta-heuristics aimed at the fixed-charge capacitated multicommodity network design formulation. The neighbourhood defines moves that explicitly take into account the impact on the total design cost of potential modifications to the flow distribution of several commodities simultaneously. Moves are identified through a shortest path-like network optimization procedure and proceed by re-directing flow around cycles and closing and opening design arcs accordingly. These neighbourhoods are evaluated and tested within a simple tabu search algorithm. Experimental results show that the proposed approach is quite powerful and outperforms existing methods reported in the literature.

Keywords : Fixed charge capacitated multicommodity network design, Meta-heuristics, Cycle-based neighbourhoods, Tabu search

Résumé

Nous proposons une nouvelle structure de voisinages pour les méta-heuristiques dédiées au problème de synthèse de réseau multiproduits avec capacités. Les mouvements ainsi définis prennent explicitement en compte l'impact sur le coût du design de modifications simultanées à la distribution de plusieurs produits. L'identification de bons mouvements se fait par l'intermédiaire d'une procédure d'optimisation basée sur une méthode de calcul de chemins les plus courts. Une solution voisine est obtenue par la re-direction de flots dans des cycles et la fermeture/ouverture subséquente d'arcs de design. Nous développons une méthode simple de recherche avec tabous pour tester cette structure de voisinage. Les résultats obtenus indiquent que cette méthode constitue la meilleure heuristique actuellement disponible pour le problème étudié.

Mots-clés : Synthèse de réseau multiproduits avec capacités, Méta-heuristiques, Voisinages par cycles, Méthodes de recherche avec tabous

3.1 Introduction

The fixed-charge capacitated multicommodity network design formulation (CMND) represents a generic model for a wide range of applications in planning the construction, development, improvement, and operation of transportation, logistics, telecommunication, and production systems (Balakrishnan, Magnanti, and Mirchandani 1997, Magnanti and Wong 1984, Minoux 1989). In these applications, multiple commodities (goods, data, people, etc.) must be routed between different points of origin and destination over a network of limited capacity. Moreover, other than the routing cost proportional to the number of units of each commodity transported over a network link, a fixed cost must be paid the first time the link is used, representing its construction (opening) or improvement costs. The objective of CMND is to identify the optimal design, that is, to select the links to include in the final version of the network in order to minimize the total system cost, computed as the sum of the fixed and routing costs, while satisfying the demand for transportation.

Fixed charge network design problems are difficult (they belong to the NP-hard complexity class, Magnanti and Wong 1984), capacitated ones even more so (Balakrishnan, Magnanti, and Mirchandani 1997) due, among other factors, to the competition of commodities for the limited network capacity and the subsequent multiplication of equivalent-cost flow distributions for a given design, and the difficulty to represent trade-offs between the arc fixed costs and capacities. As a consequence, for now, exact methods cannot solve realistically dimensioned cases (Crainic, Frangioni, and Gendron 2001, Gendron, Crainic, and Frangioni 1998, Holmberg and Yuan 2000). For capacitated problems of any interesting size, only specially tailored heuristics have proved of any help. The simplest of these are drop and add procedures based on reduced cost calculations, which determine the marginal value of including or excluding an arc from the network (Powell 1986, Koskosidis, Powell, and Solomon 1992), while the more sophisticated make use of the marginal values of paths in the network (Crainic and Rousseau 1986, Farvolden and Powell 1994, Jarvis and Mejia de Martinez 1977). These local search heuristics were applied to particular problem classes, however, and did not attempt to explore past the first local optimum. Recently, Crainic, Gendreau,

and Farvolden (2000) proposed a tabu search meta-heuristic for the path-based formulation of the problem. The method explores the space of the path-flow variables by using pivot-like moves and column generation. Even though the method produced very impressive results, its search efficiency may be limited by the fact that each move considers the impact of changing the flow of one commodity only (a pivot from one path to another). Moreover, one cannot guarantee that all paths will be considered, neither explicitly, nor implicitly.

The goal of this paper is to propose a new class of neighbourhoods for meta-heuristics aimed at the CMND. The neighbourhood defines moves that may explicitly take into account the impact on the total design cost of potential modifications to the flow distribution of several commodities simultaneously. The fundamental idea is to explore the space of the arc design variables by re-directing flow around cycles and closing and opening design arcs accordingly. Thus, compared to path-based neighbourhoods, not only the move evaluations are more comprehensive (since all commodities on a cycle are explicitly considered), but also the range of moves is broader since flow deviations are no longer restricted to paths linking origins and destinations of actual commodities. To illustrate the quality of this neighbourhood, we implement two instances in a tabu-based local search meta-heuristic. Computational experiments on problems of various sizes (up to 700 arcs and 400 commodities) show that the tabu search algorithm based on these neighbourhood structures is quite powerful, outperforming existing methods in solution quality for similar computational efforts.

The contribution of this paper is twofold. First, it introduces a new class of cycle-based neighbourhood structures for meta-heuristics aimed at the CMND, together with efficient procedures to identify good moves. Second, to demonstrate the quality of the cycle-based neighbourhoods, it proposes a very simple tabu search procedure that offers the current best heuristic solutions for the CMND.

The paper is organized as follows. In Section 3.2, we recall the formulation of the capacitated multicommodity network design problem. Section 3.3 introduces the concept of cycle-based neighbourhood structures and describes procedures to identify associated "best" moves. The tabu search algorithm is described in Section 3.4, while calibration and computational results are reported in Section 3.5. We conclude with

perspectives and a number of research avenues.

3.2 Mathematical Formulation

Let $\mathcal{G} = (\mathcal{N}, \mathcal{A})$ be a network with set of nodes $\mathcal{N}$ and set of directed arcs $\mathcal{A}$. Without loss of generality, we assume that all $(i,j) \in \mathcal{A}$ are design arcs. Let $\mathcal{P}$ denote the set of commodities to move using this network, where each commodity p has a single origin $o(p)$, a single destination $s(p)$, and a flow requirement of w^p units between its origin and destination nodes. The arc-based formulation of the CMND can then be written as follows :

$$\min z(x,y) = \sum_{(i,j)\in\mathcal{A}} f_{ij}y_{ij} + \sum_{p\in\mathcal{P}}\sum_{(i,j)\in\mathcal{A}} c_{ij}^p x_{ij}^p \tag{3.1}$$

$$\text{subject to} \sum_{j\in\mathcal{N}^+(i)} x_{ij}^p - \sum_{j\in\mathcal{N}^-(i)} x_{ji}^p = d_i^p \quad \forall i \in \mathcal{N}, \forall\, p \in \mathcal{P}, \tag{3.2}$$

$$\sum_{p\in\mathcal{P}} x_{ij}^p \leq u_{ij}y_{ij} \quad \forall (i,j) \in \mathcal{A}, \tag{3.3}$$

$$x_{ij}^p \geq 0 \quad \forall (i,j) \in \mathcal{A}, \forall p \in \mathcal{P}, \tag{3.4}$$

$$y_{ij} \in \{0,1\} \quad \forall (i,j) \in \mathcal{A} \tag{3.5}$$

where y_{ij}, $(i,j) \in \mathcal{A}$, represent the design variables that equal 1 if arc (i,j) is selected in the final design (and 0 otherwise), x_{ij}^p stand for the flow distribution decision variables indicating the amount of flow of commodity $p \in \mathcal{P}$ on arc (i,j), and

$\mathcal{N}^+(i)$: Set of outward neighbours of node i,
$\mathcal{N}^-(i)$: Set of inward neighbours of node i,
u_{ij} : Capacity of arc (i,j),
f_{ij} : Fixed cost applied on arc (i,j),
c_{ij}^p : Cost of one unit flow of commodity p on arc (i,j),

and

$$d_i^p = \begin{cases} w^p & \text{if} \quad i = o(p) \\ -w^p & \text{if} \quad i = s(p) \\ 0 & \text{otherwise.} \end{cases}$$

The objective function (3.1) accounts for the total system cost, the fixed cost of arcs included in a given design plus the cost of routing the product demand, and aims to select the minimum cost design. Constraints (3.2) represent the network flow conservation relations, while constraints (3.3) state that for each arc, the total flow of all commodities cannot exceed its capacity if the arc is opened ($y_{ij} = 1$) and must be 0 if the arc is closed ($y_{ij} = 0$). Relations (3.5) and (3.4) are the usual non-negativity and integrality constraints for decision variables.

Finally, recall that for a given design vector $\bar{y}$, the arc-based formulation of the CMND becomes a capacitated multicommodity minimum cost flow problem (CMCF) :

$$\min z(x(\bar{y})) = \sum_{p \in \mathcal{P}} \sum_{(i,j) \in \mathcal{A}(\bar{y})} c_{ij}^p x_{ij}^p \tag{3.6}$$

subject to (3.2) plus

$$\sum_{p \in \mathcal{P}} x_{ij}^p \leq u_{ij} \bar{y}_{ij} \quad \forall (i,j) \in \mathcal{A}(\bar{y}),$$

$$x_{ij}^p \geq 0 \quad \forall (i,j) \in \mathcal{A}(\bar{y}), \forall p \in \mathcal{P}.$$

where $\mathcal{A}(\bar{y})$ stands for the set of arcs corresponding to the design $\bar{y}$.

3.3 Cycle-based Neighbourhoods

Consider the solution space of the design variables y. A solution to the CMND is then an assignment $\bar{y}$ of 0 or 1 to each design variable, plus the optimal flow of the corresponding multicommodity minimum cost flow problem $x^*(\bar{y})$. Similarly, the objective function value associated to a solution $(\bar{y}, x^*(\bar{y}))$ is the sum of the fixed cost of the open arcs in $\bar{y}$ and the objective function value of the CMCF associated to $x^*(\bar{y})$:

$$z(\bar{y}, x^*(\bar{y})) = \sum_{(i,j) \in \mathcal{A}(\bar{y})} f_{ij} \bar{y}_{ij} + z(x^*(\bar{y})) \tag{3.7}$$

The classical neigbourhood used to explore such a space changes the value of one, sometimes two, variables at a time (Glover and Laguna 1997). This corresponds to

adding an arc to the design, dropping an arc from the network, or swapping the values of an open and a closed arc. Such moves have performed well in related cases such as location problems (e.g., Crainic *et al.* 1993). They fail in the case of capacitated network design problems. The main reason is that, contrary to location problems where, in most cases, paths between origins and destinations are very short (one link, usually) and each site has a significant impact on the design and corresponding flow distribution, in design problems there are many arcs, and origin-to-destination paths are many arcs long. An arc is then only one of many and its impact is generally limited. One may often reroute traffic and obtain an almost equivalent solution. Or, as illustrated by arc (a, b) in Figure 3.1, it may not even be connected to the other currently open arcs. Introducing such an arc into the network has no influence whatsoever on the current solution. One needs, therefore, moves that acknowledge that commodities move on paths and that open and close several arcs simultaneously. The goal of this paper is to introduce such a neighbouring structure. But first, a few basic graph definitions :

1. A simple directed *path* from node i_0 to i_n is a sequence of closed or opened arcs $\{(i_0, i_1), (i_1, i_2), ..., (i_{n-1}, i_n\}$ such that the origin node of each arc is the destination node of the preceding arc in the sequence, and $i_0, ...i_n$ are all distinct nodes. In Figure 3.1, $\{(h, f), (f, a), (a, b), (b, c), (c, s)\}$ is a path.
2. A simple *chain* is similar to a path except that arcs are not restricted to follow the same direction. In Figure 3.1, $\{(a, b), (b, c), (c, d)\}$ represents a chain.
3. A (simple) *cycle* is a closed chain, such as $\{(a, b), (b, c), (c, d), (d, e), (e, f), (f, a)\}$ in Figure 3.1.

3.3.1 Neighbourhood Definition and Exploration

The fundamental idea behind the new neighbourhood class is that one may move from one solution to another by

1. Identifying two points in the network together with two paths connecting these points, thus closing a cycle;
2. Deviating the total flow from one path to another such that at least one currently

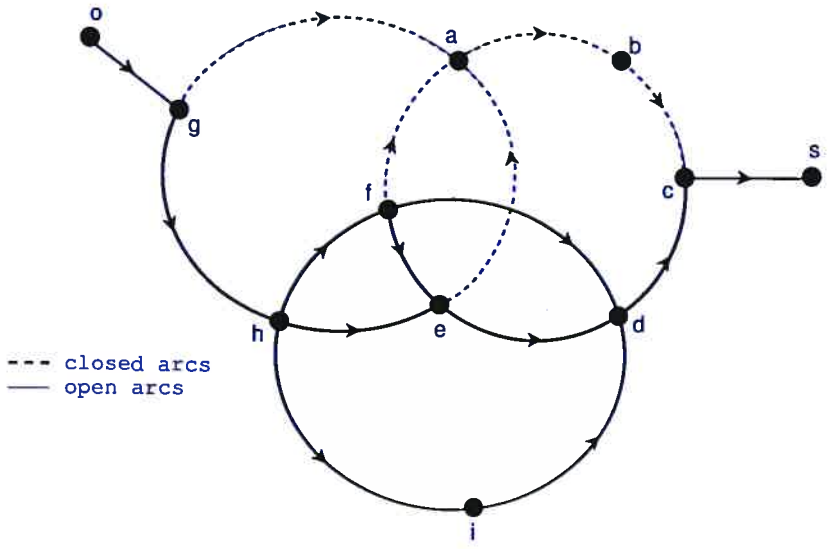


FIG. 3.1 – Example of Network, Paths, and Cycles

open arc becomes empty;

3. Closing all previously open arcs in the cycle that are empty following the flow deviation and, symmetrically, opening all previously closed arcs that now have flow.

To illustrate, paths $\{(f,e),(e,d),(d,c)\}$ and $\{(f,a),(a,b),(b,c)\}$ in the partial network depicted in Figure 3.1 form a cycle and flow from the former may be deviated to the latter. Three arcs will thus be open, (f,a), (a,b), and (b,c) and, provided sufficient flow may be deviated, at least one previously open arc will be closed. The general neighbourhood structure we propose for the CMND may then be written as

$\mathcal{V}(\bar{y}) = \{\ y$: obtained from $\bar{y}$ by complementing the status of a number of arcs following the deviation of flow in a given cycle in $\mathcal{A}(\bar{y})\ \}$

Such neighbourhoods are huge and their explicit and exhaustive exploration is not practical in most situations. Moreover, the complete evaluation of any design modification involves the resolution of a capacitated multicommodity network flow problem, which rapidly becomes extremely computationally intensive. Thus, in order to select the best move out of a given solution, one cannot simply identify all neigh-

bours explicitly, evaluate them, and retain the best one. A more efficient procedure must be implemented that 1) avoids the complete evaluation of every examined move and 2) generates a limited number of cycles that include the "good" moves. One such procedure is presented in the following and it is implemented and tested in Section 3.4.

Note that not all cycles are of equal interest. We seek, in particular, moves that modify the status of several arcs and that lead to a significant modification of the flow distribution. Therefore, moves that close at least one arc and open new paths for a group of commodities appear attractive.

To close an arc (i,j), one must be able to deviate all its flow. Let the *residual capacity* of a cycle denote the maximum flow one can deviate around the cycle. The residual capacity of any cycle that includes (i,j) must then be at least as large as $\sum_{p\in\mathcal{P}} x_{ij}^p$, the total flow on arc (i,j). Consequently, cycles of interest to us display a residual capacity equal to one of the values in Γ defined as the set of the total (strictly positive) volumes on the open arcs of the corresponding network :

$$\Gamma(\bar{y}) = \left\{ \sum_{p\in\mathcal{P}} x_{ij}^p > 0 \ : \ (i,j) \in \mathcal{A}(\bar{y}) \right\}. \tag{3.8}$$

Cycles have thus to be identified on *residual networks* of $\mathcal{A}(\bar{y})$ defined according to $\Gamma(\bar{y})$ and the one leading to the network modification that yields the largest improvement (smallest deterioration, eventually) in equation (3.7) corresponds to the best move in the neighbourhood of $\bar{y}$. But, again, finding all such cycles and explicitly evaluating the corresponding moves is impractical and some heuristic must be used. We therefore propose a network optimization procedure to implicitly evaluate a large number of cycles according to criteria that approximate the evaluation criterion defined by equation (3.7). We thus progressively build a set of good candidate neighbours (cycles) among which the best move is then selected.

Let $\mathcal{C} \subseteq \mathcal{A}$ denote a set of *candidate* links, where each arc $(i,j) \in \mathcal{C}$ will be considered as the starting point of cycles. Let $\mathcal{C}(\gamma) \subseteq \mathcal{C}$ include all arcs $(i,j) \in \mathcal{C}$ that can support a movement of γ units of flow. A closed arc may belong to $\mathcal{C}(\gamma)$ only if its capacity is at least γ. For an open arc, either its flow or its residual capacity must

be at least γ units. The heuristic procedure we propose to explore the neighbourhood $\mathcal{V}(\bar{y})$ of a given solution $\bar{y}$ may be synthesized as follows :

- Build sets $\mathcal{C}(\bar{y})$ and $\Gamma(\bar{y})$;
- For each $\gamma \in \Gamma(\bar{y})$
  - Build the γ-residual network as indicated in Section 3.3.2 ;
  - For each arc $(i,j) \in \mathcal{C}(\gamma)$, find the lowest cost cycle by using the network optimization procedure of Section 3.3.3 ;
- Select and implement the best move ;
- Evaluate new solution.

This procedure is embedded in the tabu-based local search procedure described in Section 3.4. First, however, we define γ-residual networks and describe the network optimization procedure used to find low-cost cycles in residual networks.

3.3.2 The γ-Residual Network

The objective is to build a network such that, given a flow value γ, the network optimization procedure of Section 3.3.3 may 1) identify low cost cycles that support the deviation of at least γ units of flow and 2) explicitly consider the impact of a potential closure or opening of an arc due to the deviation of flow.

To build the *γ-residual network* corresponding to a flow variation of γ units, replace each arc (i,j) of the original network by at most two arcs $(i,j)^+$ and $(j,i)^-$.

Arc $(i,j)^+$ is included only if an additional amount of γ units of flow may pass on arc (i,j), that is, if its residual capacity $u_{ij} - \sum_{p \in \mathcal{P}} x_{ij}^p$ is greater or equal to γ. The cost c_{ij}^+ associated to $(i,j)^+$ approximates the cost of routing γ units of additional flow on arc (i,j). It equals the average commodity routing costs on the arc, plus the fixed cost if it is currently closed :

$$c_{ij}^+ = \frac{\sum_{p \in \mathcal{P}} c_{ij}^p}{\mid P \mid} \gamma + f_{ij}(1 - (\lceil \min(1, \sum_{p \in \mathcal{P}} x_{ij}^p) \rceil)).$$

Symmetrically, arc $(j,i)^-$ is not included in the γ-residual network if the total flow on arc (i,j), $\sum_{p \in \mathcal{P}} x_{ij}^p$, is less than γ. Otherwise, we associate to $(j,i)^-$ the cost c_{ji}^- that

approximates the value of a reduction of γ units of the total flow (all commodities) currently using arc (i,j). This approximation is computed as the weighted average of the routing costs of the commodities currently using arc (i,j). The fixed cost of (i,j) is then subtracted if the reduction of γ units of flow leaves the arc empty :

$$c_{ji}^{-} = \begin{cases} -\frac{\sum_{p\in\mathcal{P}} c_{ij}^{p} x_{ij}^{p}}{\sum_{p\in\mathcal{P}} x_{ij}^{p}}\gamma - f_{ij} & \text{if } \sum_{p\in\mathcal{P}} x_{ij}^{p} = \gamma \\ -\frac{\sum_{p\in\mathcal{P}} c_{ij}^{p} x_{ij}^{p}}{\sum_{p\in\mathcal{P}} x_{ij}^{p}}\gamma & \text{if } \sum_{p\in\mathcal{P}} x_{ij}^{p} > \gamma \end{cases}$$

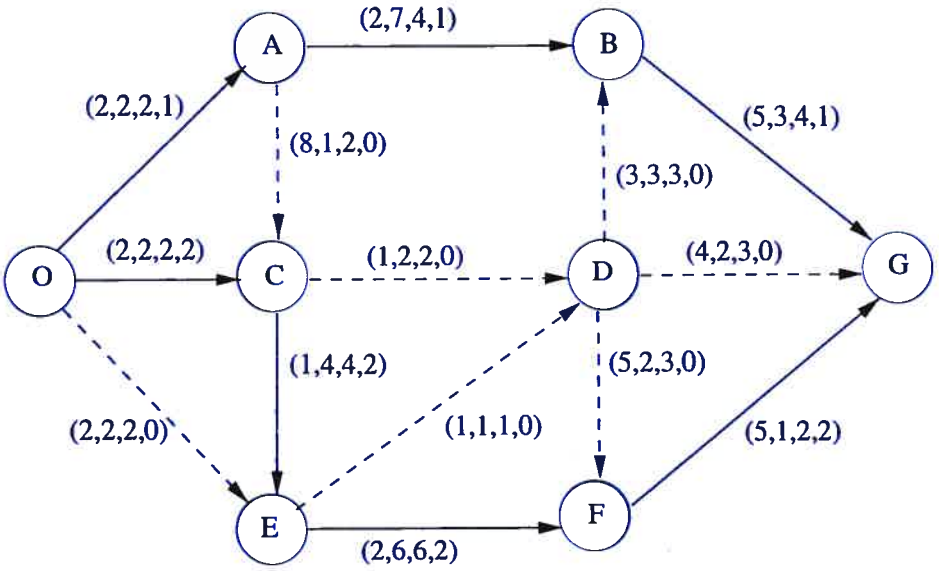
To illustrate this procedure, consider the network of Figure 3.2. In this example, arcs are labeled by their fixed cost, routing cost (only one commodity is considered), capacity, and total flow, respectively. We build a 2-residual network and look for the lowest cost cycle that passes through arc (C,D).

Two directed cycles pass through arc (C,D) in the 2-residual network of Figure 3.3 : $(D,F),(F,E),(E,C),(C,D)$ and $(D,G),(G,F),(F,E),(E,C),(C,D)$. The latter has the lowest cost with value -17. This cost represents the net change in the objective function of the design formulation when moving 2 units of flow from path $(C,E),(E,F),(F,G)$ to path $(C,D),(D,G)$, opening arcs (C,D) and (D,G), and closing arcs $(C,E),(E,F)$ and (F,G) in the original network. This neighbour is thus reached by a complex move that involves opening two arcs $((C,D),(D,G))$ and closing three others $((C,E),(E,F),(F,G))$.

3.3.3 Heuristic to Identify Low Cost Cycles

The enumeration of all cycles passing through a given arc in a γ-residual network is expensive. Since one is interested in the lowest cost cycle only, one would like to find this particular cycle without enumerating all the others. A network optimization procedure is proposed for this purpose.

The main idea is to find for a given arc (i,j) the shortest path from j to i to complete the cycle. Yet, since the γ-residual network may contain negative cost directed cycles, finding a shortest path is NP-hard (Ahuja, Magnanti, and Orlin 1993). Consequently, we do not attempt to solve the problem exactly. We rather develop a



(fixed cost, routing cost, capacity, flow)

FIG. 3.2 – Network Example

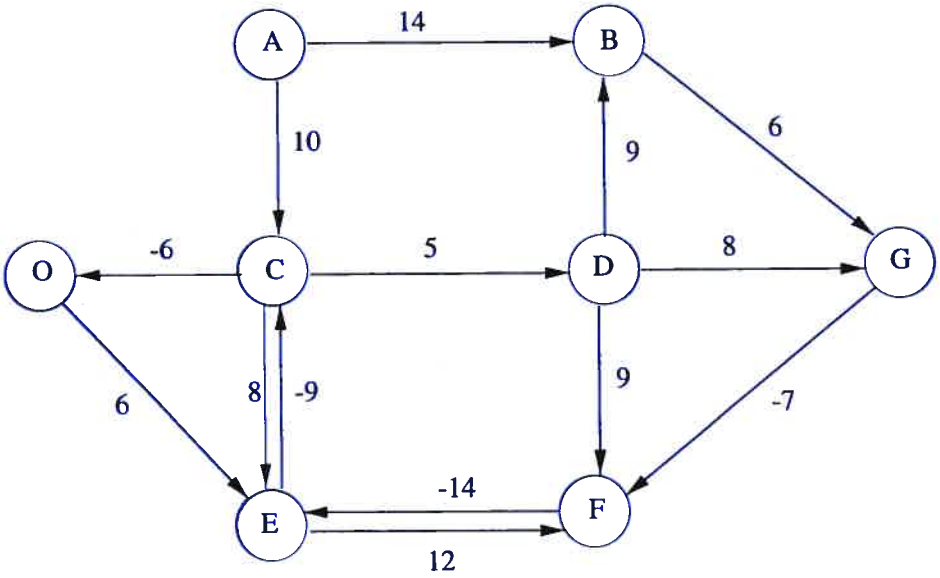


FIG. 3.3 – Associated 2-Residual Network

heuristic based on a modification of the shortest path label-correcting algorithm that avoids getting trapped in negative directed cycles and allows to find low-cost cycles.

Label-correcting (Ahuja, Magnanti, and Orlin 1993) is one of the algorithms designed to find shortest paths between a source node and all other nodes in a network which contains no negative directed cycles. The algorithm starts by labelling each node j of the network with $d(j)$, the current distance (cumulated cost) in the shortest path tree from the source to the node. Nodes are also labelled with their respective predecessors in the current shortest paths, $pred(j)$, in order to be able to retrace the shortest paths. The algorithm then proceeds by decreasing, at every iteration, the distance label of a node j in order to satisfy the shortest path optimality conditions

$$d(j) \leq d(i) + c_{ij} \quad \forall (i,j) \in \mathcal{A},$$

where c_{ij} denotes the cost of arc (i,j).

Label-correcting algorithms operate by maintaining a LIST of nodes i that may be the origin of links for which the optimality condition is not yet satisfied. At each iteration, one node i is removed from LIST and the nodes $j \in \mathcal{N}^+(i)$ are scanned. If $d(j) > d(i) + c_{ij}$, $d(j)$ is set to $d(i) + c_{ij}$ and j is added to LIST (if not already included). The algorithm terminates when LIST is empty.

In the presence of negative-cost directed cycles, the algorithm gets trapped and keeps cycling. To avoid this, we modify the label-correcting algorithm to explicitly verify if a scanned node j already belongs to the current shortest path from the source node to its predecessor node i, in which case the label of node j is not modified (we forbid node j to be labeled from node i). The label-correcting-based heuristic then becomes :

$d(s) = 0$, $pred(s) = 0$, $d(j) = \infty$ for each $j \neq s$; LIST $= \{s\}$

While LIST $\neq \phi$

Remove an element i from LIST

For each node j in its forward star $\mathcal{N}^+(i)$

If $(d(j) > d(i) + c_{ij})$ and (j is not in the path from s to i)

- $d(j) = d(i) + c_{ij}$;

- $pred(j) = i$;
- If $j \notin$ LIST then add j to LIST.

The path generated by this heuristic is not necessarily the shortest path. Computational results show, however, that the heuristic produces very good cycles.

3.4 Tabu Search with Cycle-based Neighbourhoods

Tabu search is an iterative meta-heuristic (Glover 1986, 1989, 1990) that has consistently been successful in addressing hard optimization problems (Glover and Laguna 1997), including the CMND (Crainic, Gendreau, and Farvolden 2000). To evaluate the concepts introduced in the previous section, we developed a simple tabu search-based local search procedure that integrates two versions of the cycle-based neighbourhood : one that considers the flow of all commodities when determining cycles and a second one that refines the search by implementing moves resulting from the deviation of the flow of one commodity only at a time. We first present the basic tabu search framework we use. Then, we examine how the procedure deals with infeasible solutions and how the search is intensified following a move to a "good" solution.

3.4.1 The Tabu Search Procedure

Following an initialization phase, the tabu search procedure explores the y solution space using a simple local search framework based on the neighbourhood defined in Section 3.3. At each iteration, the best non-tabu move is determined and implemented regardless whether it improves the overall solution or not. A short-term tabu memory is used to record characteristics of visited solutions to avoid cycling. When a particularly good solution is encountered, the search is intensified using a particular implementation of cycle-based neighbourhoods that considers the flow distribution of one commodity only. A solution is considered particularly good when it either improves the objective value of the best known solution or is within a pre-defined percentage of this value. The method terminates whenever a predefined stopping criterion (number of iterations, CPU time, etc.) is met.

To select the "best" move in the neighbourhood of a given solution $\bar{y}$, the procedure first determines the set of the flow deviation values $\Gamma(\bar{y})$, as defined by (3.8), and the set of candidate arcs $\mathcal{C}$. In the current implementation, $\mathcal{C}$ is restricted to a random subset of closed arcs. Then, following the framework detailed in Section 3.3.1, a γ-residual network is built for each value $\gamma \in \Gamma(\bar{y})$ (Section 3.3.2), and a low-cost cycle is found for each arc $(i,j) \in \mathcal{C}(\gamma)$ on the corresponding γ-residual network (Section 3.3.3). The lowest overall cycle, for all $(i,j) \in \mathcal{C}(\gamma)$ and $\gamma \in \Gamma(\bar{y})$, is then considered as the "best" *local search move* from the current solution, and arcs are open and closed accordingly. An exact evaluation of the new design is then performed by solving exactly the associated capacitated multicommodity network formulation. This evaluation may result in some open arcs carrying no flow. One may then *trim* - close - these arcs, further reducing the associated solution value. Notice, however, that the trim procedure modifies the basic move definition as well as the search trajectory (Section 3.5.1). An outline of the tabu search procedure follows.

Initialization : Generate an initial feasible solution and set *BestSolution* and *CurrentSolution* to its objective value.

Main local search loop

While a stopping criterion is not met

- Determine sets $\mathcal{C}$, $\Gamma(\bar{y})$, and $\mathcal{C}(\gamma)$ for the current solution $\bar{y}$;
- Determine the best cycle over all $(i,j) \in \mathcal{C}(\gamma)$, $\gamma \in \Gamma(\bar{y})$;
- Move to the new solution by opening and closing the appropriate arcs; Determine the solution value of the new solution by solving exactly the associated capacitated multicommodity network flow problem;
- Assign a tabu status to each complemented arc;
- Trim;
- If the solution is infeasible, perform a *restoration* phase;
- Perform an *Intensification* phase if current solution is "good" :

$$\frac{CurrentSolution - BestSolution}{BestSolution} \leq IntensGap;$$

- If $CurrentSolution < BestSolution$ update *BestSolution*.

Short-term tabu memory is used to prevent the search from cycling. Thus, the arcs that are opened or closed receive a tabu status that forbids the reversal of the move for a given number of iterations. The tabu memory is updated following local search and intensification moves.

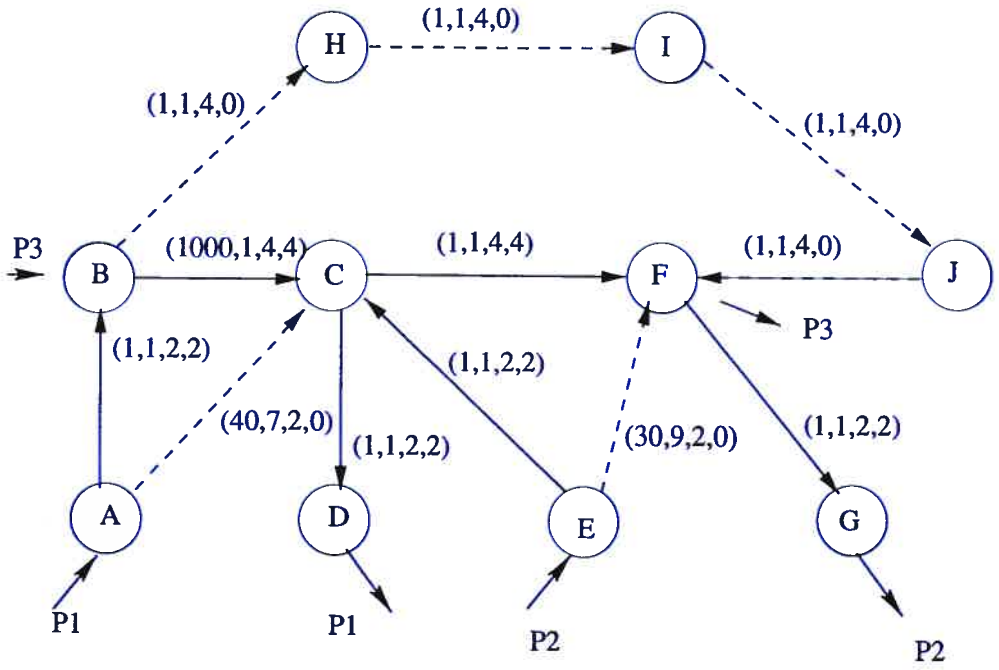
An initial feasible solution is produced by opening all arcs and solving the corresponding flow problem. All arcs with flow are then considered open, while all unused arcs are closed. An intensification phase is then performed until negative cycles are no longer detected for any commodity.

3.4.2 Restoration from Infeasible Moves

The solution produced by a local search move might be infeasible. This follows from the fact that commodities are aggregated when local search moves are determined on γ-residual networks. Consequently, in the new network configuration that follows the closing and opening of arcs in the original network, commodities might follow different paths than the expected ones (i.e., those in the local search cycle-move).

To illustrate, consider the partial network displayed in Figure 3.4 where three commodities p_1, p_2, and p_3 share the network. Let (x, y, z, t) denote respectively the fixed cost, routing cost, capacity, and flow on each arc and suppose that $(A, D, 2)$, $(E, G, 2)$, and $(B, F, 2)$ represent the origin, destination, and demand of commodity p_1, p_2, and p_3, respectively. The current solution routes 2 units of flow of commodity p_1 on path $(A, B), (B, C), (C, D)$, 2 units of flow of commodity p_2 on path $(E, C), (C, F), (F, G)$, and 2 units of flow of commodity p_3 on path $(B, C), (C, F)$. The objective function value associated to the current solution equals 1021.

In this example $\Gamma = \{2, 4\}$. Assume all closed arcs are in $\mathcal{C}$ and an empty tabu list. Then, the lowest cost cycle in the 4-residual network of Figure 3.5 shows an improvement of -989. In the 2-residual network (not shown), the lowest cost cycle yields a deterioration of the objective function. Consequently, the procedure would retain the cycle $(B, H), (H, I), (I, J), (J, F), (F, C), (C, B)$ and would move to the best neighbour by opening arcs (B, H), (H, I), (I, J), and (J, F) and closing the arcs without flow (B, C) and (C, F). The resulting network is infeasible since the demands



(fixed cost, routing cost, capacity, flow)

FIG. 3.4 – Example of Infeasible Move : Original Network

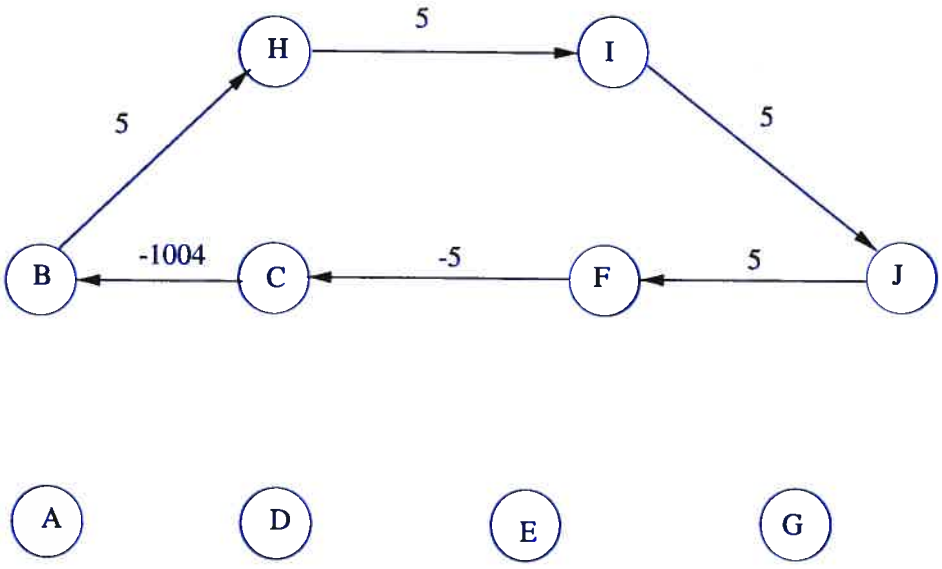


FIG. 3.5 – Example of Infeasible Move : Residual Network

of commodities p_1 and p_2 may no longer be satisfied.

To detect infeasible solutions, artificial arcs between the origin and destination nodes of each commodity may be added to the network. These arcs receive capacities equal to the demand of the associated commodities and high routing costs to ensure that an artificial arc will bear flow only if the solution is infeasible. Then, when the solution produced by a local search move is infeasible, a restoration phase is undertaken in order to reach a solution in the feasible domain. This phase is similar to the neighbourhood local search, except that the set $\mathcal{C}$ is made exclusively of artificial arcs with a positive flow, while the set Γ is restricted to the flow of commodities routed on these artificial arcs.

3.4.3 Intensification

The intensification phase is called each time a local search move yields a "good" solution, that is, a solution that improves the best overall solution or is close to it. This phase searches to improve the solution further by iteratively modifying the flow distribution of one commodity at a time.

The neighbourhood definition of Section 3.3 is adapted for the intensification phase : Moves are detected by letting the flow of one commodity move around negative directed cycles while the flow of the other commodities is kept fixed. The set Γ is instantiated for each commodity (denoted Γ^p) to contain the flow of the particular commodity that exists on each arc of the network. Moreover, only improving moves are accepted during this phase to ensure that the chosen neighbour will always yield a feasible solution better than the current one. An outline of the intensification phase follows.

Intensification Phase

Repeat until no negative cycle is detected in any γ-residual network

For each commodity $p \in \mathcal{P}$ and flow value $\gamma \in \Gamma^p$

- Build the γ-residual network
- Find a low cost cycle for each arc $(i,j) \in \mathcal{C}(\gamma),\ \gamma \in \Gamma^p$
- Select the best overall cycle;

  Denote by γ_{best} the associated volume of deviated flow;

- If the selected cycle is improving (negative cost)
  1. Update the solution :
     - Modify the flow around the selected cycle by a quantity γ_{best} ;
     - Open/close appropriate arcs ;
     - Update the cost of the current solution by adding the cost of the cycle (without solving the CMCF).
  2. Assign a tabu status to each complemented arc.

Solve exactly the CMCF associated to the current design.

3.5 Computational Results

Experiments have been performed to evaluate the performance of the simple tabu search meta-heuristic and, more generally, of the cycle-based neighbourhood concept we propose. To ensure meaningful comparisons, we experiment on the same two sets of problem instances used by Crainic, Gendreau, and Farvolden (2000).

The computer code is written in C++. The exact evaluation of the capacitated multicommodity network flow problems is performed using the linear programming solver of CPLEX 6.5 (1999). Unless indicated otherwise, tests were conducted on a Sun Ultra-60/2300 workstation with 2 Gigabyte of RAM, operating under Solaris 2.6. Computing times are reported in seconds.

3.5.1 Calibration

A calibration phase was conducted to determine appropriate values of key parameters of the tabu search meta-heuristic :

- The size of the neighbourhood set explored at each iteration. This size depends on the dimension of the candidate set $\mathcal{C}$ defined in the current implementation as a percentage of closed arcs randomly selected. Two values, 50% and 70% of the closed arcs, have been tested.
- The length of the tabu tenure of arcs opened and closed following a local search or an intensification move. Four values - 1, 2, 3, and 5 - were considered initially. The 1 and 5 values were rapidly dropped, however, since cycling was observed

for a tabu tenure length of 1, while for a value equal to 5 the quality of solutions started to decrease.

- The threshold used to determine a "good" solution : Values of *IntensGap* equal to 7%, 9%, and 11% of relative improvement were tested.

Calibration experiments were conducted on the same problem instances used by Crainic, Gendreau, and Farvolden (2000) to calibrate their path-based tabu search meta-heuristic. The 10 problems cover network sizes from 100 to 700 design arcs and from 10 to 400 commodities. They also display relatively high fixed costs compared to routing costs and are tightly capacitated. Moreover, since a random number generator controls the selection of arcs in $\mathcal{C}$, each run was repeated three times.

Each parameter combination was ranked for each problem instance according to the average gap (over three runs) relative to the best known solution (that of the branch-and-bound procedure of CPLEX 6.5, when available, or that obtained by Crainic, Gendreau, and Farvolden (2000, otherwise). A score of 10, 9, 8, .. , 2, 1 is assigned to each of the first ten places, respectively. The performance of each parameter setting is then aggregated over all 360 runs (10 problems tested 3 times for 12 parameter settings) : scores are summed up, while average gaps and CPU solution times are averaged. Table 3.1 displays these aggregated results for each parameter combination. The first column holds the parameter setting, the second and third columns present the global average gaps and CPU times, respectively, while the last column displays the total score.

The results in Table 3.1 display two parameter settings that achieve the same highest aggregated score. The final choice was then made on the other performance measures and the parameter combination $\mathcal{C}$=50%, tabu tenure= 2, *IntensGap*= 9% was selected since it offers the lowest global average gap and CPU time. Two additional comments are prompted by the results displayed in Table 3.1. First, that global averages may hide important variations and may be misleading. Thus, the lowest average gap appears for the fourth ranking parameter setting (70%, 2 and 7%). Secondly, one notices that more than one parameter setting leads to very good performances, which is an indication of the robustness of the search.

TAB. 3.1 – Parameter Setting Performances

Parameter Setting	Average Gap	Average CPU	Score
50%, 2, 0.09	1.93%	10814.34	68
70%, 2, 0.11	2.07%	13068.72	68
70%, 2, 0.09	2.09%	13011.97	63
70%, 2, 0.07	1.84%	13168.14	62
70%, 3, 0.07	1.96%	12427.75	59
50%, 3, 0.07	2.27%	10984.97	46
50%, 2, 0.07	2.30%	11480.44	45
70%, 3, 0.09	2.62%	12826.49	40
70%, 3, 0.11	2.95%	13528.20	35
50%, 3, 0.11	2.83%	11530.40	26
50%, 2, 0.11	2.72%	10929.73	24
50%, 3, 0.09	2.81%	11172.91	14

To complete the calibration phase, we examined the impact of the intensification phase and the trim procedure on the solution quality. We started the investigation by running five problems using all four possible combinations of including or not the two procedures. Since the results were similar on all five problems, we decided not to continue the investigation. Figure 3.6 illustrates the behaviour of the four versions of the tabu search procedure on two problem instances, 20,300,200,F,L and 20,300,40,F,T (identified by the number of nodes, design arcs, commodities, a letter indicating high fixed costs, and rather loose and tight capacities, respectively). This behaviour is typical of the general method performance and thus, in order not to overload the paper, the other graphs are not shown. The results clearly indicate that including the intensification and trim procedures is beneficial for the quality of the search. Consequently, all subsequent experiments have included the two procedures.

3.5.2 Result Analysis

To evaluate the performance of the tabu search algorithm proposed in this paper, we compare its output to the optimal solution obtained using the branch-and-bound algorithm of CPLEX 6.5 (1999), as well as to the results of the TABU-PATH procedure presented by Crainic, Gendreau, and Farvolden (2000), which was found to be superior

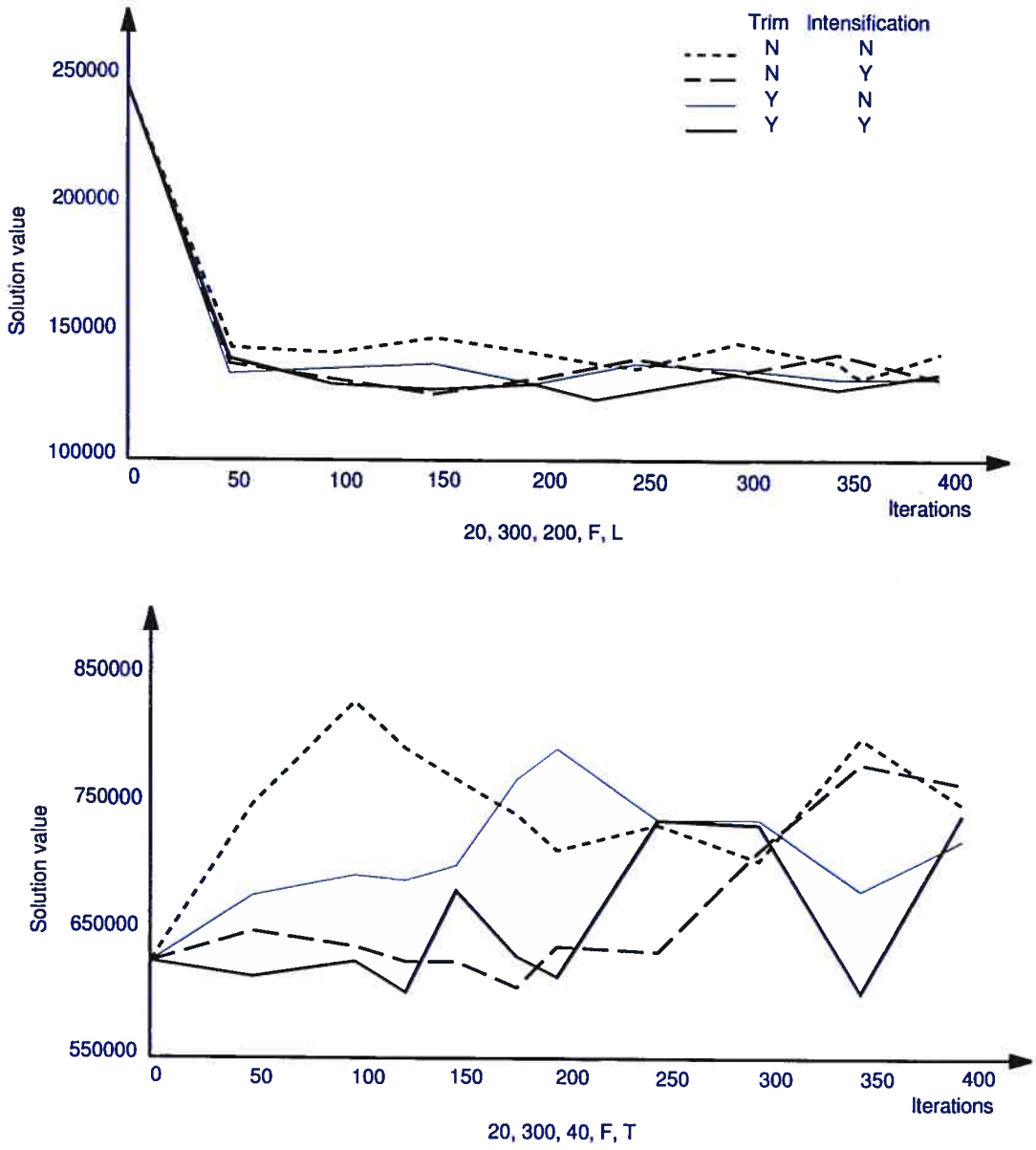


FIG. 3.6 – Calibration for Intensification and Trim

to both classical greedy descent and resource decomposition (Gendron and Crainic 1996).

The same two data sets used by Crainic, Gendreau, and Farvolden (2000) are also used in this paper. Detailed results for the first set, denoted **C**, are presented in this section. Results for the second set, denoted **R**, are detailed in the appendix and are summarized in the following. Problems in both sets are general transshipment networks with no parallel arcs. Each commodity corresponds to a single origin-destination pair. On each arc, routing costs are the same for all commodities. Problem instances have been generated to offer for each network size a variety of fixed cost to routing cost ratios and capacity to demand ratios. A detailed description of problem instances is given in Crainic, Frangioni, and Gendron (2001 ; see also Gendron and Crainic 1994, 1996). The problem generators as well as the problem instances can be obtained from the authors.

Tables 3.2 and 3.3 display results for the first set, denoted **C**. Problems are identified in the first column by the number of nodes, arcs, and commodities, as well as two letters summarizing the fixed cost and capacity information : a relatively high or low fixed cost relative to the routing cost is signaled by the letter **F** or **V**, respectively, while letters **T** and **L** indicate respectively if the problem is tightly or somewhat loosely capacitated compared to the total demand. The OPT column corresponds to the solution of the the branch-and-bound algorithm solved using CPLEX 6.5 (1999) on one 400MHz processor of a 64-CPU Sun Enterprise 10000 with 64 Gigabyte of RAM, operating under Solaris 2.7. A limit of 10 hours was imposed. An **X** indicates that the procedure has failed to produce a feasible solution within this time limit, while a **t** indicates that the procedure stopped due to a time limit condition.

Column TABU-PATH holds the results found by the path-based tabu search of Crainic, Gendreau, and Farvolden (2000) on the same workstations. For a valid comparison between the two approaches, two results are displayed for the cycle-based meta-heuristic. The TABU-CYCLE column displays solutions found by our approach using the same CPU time reported for the path-based method. Longer runs were performed to further illustrate and analyze the behaviour of the algorithm we propose. Column TC(400) displays computational results after 400 iterations. For the three

TAB. 3.2 – Computational Results, C problems

PROBLEM	OPT	TABU-PATH	GAP	TABU-CYCLE	GAP	TC(400)	GAP
25,100,10,V,L	14712	14712	0%	14712	0%	14712	0%
	(0.36)	(5.60)		(5.60)		(48.88)	
25,100,10,F,L	14941	15889	6.34%	14941	0%	14941	0%
	(53.64)	(8.37)		(8.37)		(53.77)	
25,100,10,F,T	49899	51654	3.52%	50767	1.74%	49899	0%
	(40.58)	(17.10)		(17.10)		(51.21)	
25,100,30,V,T	365272	365272	0%	365385	0.03%	365385	0.03%
	(16.62)	(16.57)		(16.57)		(223.67)	
25,100,30,F,L	37055	38804	4.72%	38679	4.38%	37583	1.42%
	(1727.96)	(33.01)		(33.01)		(215.38)	
25,100,30,F,T	85530	86445	1.07%	86296	0.90%	86296	0.90%
	(534.18)	(71.84)		(71.84)		(224.64)	
20,230,40,V,L	423848	425046	0.28%	425091	0.29%	424778	0.22%
	(10.43)	(71.29)		(71.29)		(370.33)	
20,230,40,V,T	371475	371816	0.09%	372583	0.30%	371893	0.11%
	(52.37)	(90.28)		(90.28)		(435.61)	
20,230,40,F,T	643036	644172	0.18%	646786	0.58%	645812	0.43%
	(671.76)	(121.79)		(121.79)		(423.32)	
20,300,40,V,L	429398	429912	0.12%	429837	0.10%	429535	0.03%
	(3.76)	(71.05)		(71.05)		(611.5)	
20,300,40,F,L	586077	589190	0.53%	593323	1.24%	593322	1.24%
	(145.55)	(113.44)		(113.44)		(581.94)	
20,300,40,V,T	464509	464509	0%	466525	0.43%	464724	0.05%
	(83.88)	(145.33)		(145.33)		(589.64)	
20,300,40,F,T	604198	606364	0.36%	609285	0.84%	607100	0.48%
	(59.88)	(123.42)		(123.42)		(560.38)	
20,230,200,V,L	94386	122592	29.88%	103930	10.11%	98995	4.88%
	(t)	(504.50)		(504.50)		(2663.24)	
20,230,200,F,L	141737.4	188590	33.06%	154404	8.94%	146535	3.38%
	(t)	(491.63)		(491.63)		(2718.31)	
20,230,200,V,T	97914	118057	20.57%	108274	10.58%	104752	6.98%
	(t)	(548.36)		(548.36)		(2565.71)	
20,230,200,F,T	137271	182829	33.19%	153455	11.79%	147385	7.37%
	(t)	(889.69)		(889.69)		(3120.14)	
20,300,200,V,L	74972.4	88398	17.91%	81628	8.88%	80819	7.80%
	(t)	(982.21)		(982.21)		(4086.83)	
20,300,200,F,L	117306	151317	28.99%	129600	10.48%	123347	5.15%
	(t)	(1316.75)		(1316.75)		(4367.88)	
20,300,200,V,T	74991	82724	10.31%	80510	7.36%	79619	6.17%
	(t)	(938.29)		(938.29)		(3807.93)	
20,300,200,F,T	108252	135593	25.26%	122547	13.21%	114484	5.76%
	(t)	(1065.88)		(1065.88)		(4657.54)	

meta-heuristic results, the next column displays the optimality GAP relative to the branch-and-bound solution. When branch-and-bound has failed to identify a feasible solution, as in the last part of Table 3.3, the gap relative to the path-based tabu search is displayed instead. For all methods, the figures in parentheses represent total computational time in CPU seconds on the appropriate computers.

A first conclusion that emerges from the two tables is that fixed cost, capacitated, multicommodity network design problems are indeed difficult to solve, as indicated by the performance of a state-of-the-art mixed integer programming solver. Experimental results also indicate clearly that TABU-CYCLE outperforms TABU-PATH for an equivalent computational effort in almost all experiments. Thus, out of 43 problem instances, only for 9, relatively small, problems are the solutions of TABU-PATH better than those of TABU-CYCLE. Most of these problems are "easy", however. That is, problem dimensions are sufficiently small for an advanced branch-and-bound method to find the optimal solution in very reasonable computing times. There is no need for meta-heuristics using complex neighbourhoods to address such problems. One reaches very rapidly, however, problem sizes and characteristics for which comparisons and the use of advanced meta-heuristics is meaningful and necessary.

The superiority of TABU-CYCLE appears thus to be greater when fixed costs are high : the optimality gap of TABU-CYCLE for these problems is at most 14%, while it reaches 33% for TABU-PATH. The average percentage improvement of TABU-CYCLE compared to TABU-PATH is around 3.36%, with a maximum improvement of 18.13%. These results are a first indication of the effectiveness of the cycle-based neighbourhood structures to generate good solutions for the CMND.

The results displayed in Tables 3.2 and 3.3 also support the hypothesis that our algorithm may identify higher quality solutions given longer search times. Using 400 iterations, improved solutions were found for 35 out of the 41 problems (2 solutions were already optimal). Moreover, the benefits obtained from increasing the computational effort appear to become greater for larger instances. Thus, for example, for problems with 200 commodities the maximum optimality gap is decreased from 13.21% to 7.80%. Compared to TABU-PATH, the average improvement obtained by TC(400) is increased to 4.93% and the maximum improvement is increased to 22.30%. The

results discussed in the rest of the section are obtained by using 400 iterations of the cycle-based tabu search.

A more in-depth study of the performance of the proposed methodology was conducted using the second set of problem instances, denoted **R**. These problems were generated to facilitate the study of the impact of particular problem characteristics on algorithmic performance. The number of nodes, design arcs, and commodities are systematically varied, one at a time, and for each of these networks, nine problem instances were created by combining three levels of fixed cost ratios and three levels of capacity ratios. The *fixed cost ratio* is computed as $|\mathcal{P}| \sum_{(ij)\in\mathcal{A}} f_{ij} / \sum_{p\in\mathcal{P}} w^p \sum_{(ij)\in\mathcal{A}} c_{ij}^p$, and the three values considered are F01 = 0.01, F05 = 0.05, and F10 = 0.10 corresponding to continuously higher levels of fixed costs compared to the routing costs. The *capacity ratio* is is computed as $|\mathcal{A}| \sum_{p\in\mathcal{P}} w^p / \sum_{(ij)\in\mathcal{A}} u_{ij}$, and the values considered are C1 = 1, C2 = 2, and C8 = 8, indicating that the total capacity available becomes increasingly tighter relatively to the total demand. Thus, the easiest problems generally have F01 and C1 ratios, while the most difficult ones have F10 and C8 characteristics. Detailed results are reported in the appendix. Aggregated statistics are used in the following to support the discussion.

Table 3.4 displays the distribution of the optimality gap relative to CPLEX branch-and-bound for the two sets of problem instances, obtained by the cycle-based tabu search after 400 iterations. The first column identifies the problem set, the second the total number of instances in the set, the third column indicates the number of problems where branch-and-bound did not find a feasible solution (in 10 hours), the forth indicates the number of optimal solutions found by the meta-heuristic, while the fifth displays the number of problems where the heuristic solution is better than the one found by branch-and-bound after 10 hours of computation. The next six columns correspond to the indicated gap intervals. For problems in set **C**, the optimality gap never exceed 8% and is often much smaller than 2%, for an average within 2.10% of the best solutions found by branch-and-bound. For the 153 problems of set **R**, 28 optimum solutions were found, 110 solutions are within 8%, 79 of which are within 4%, while only 7 solutions display a gap greater than 10%. The average gap over all **R** problems is 2.97%. This constitutes a very good performance, especially given the

TAB. 3.3 – Computational Results, C problems

PROBLEM	OPT	TABU-PATH	GAP	TABU-CYCLE	GAP	TC(400)	GAP
100,400,10,V,L	28423	28485	0.22%	28708	1.00%	28677	0.89%
	(84.81)	(32.66)		(32.66)		(336.34)	
100,400,10,F,L	24436	24912	1.95%	24576	0.57%	23949	- 1.99%
	(t)	(33.00)		(33.00)		(306.79)	
100,400,10,F,T	66364	71128	7.18%	68004	2.47%	67014	0.98%
	(t)	(81.23)		(81.23)		(626.46)	
100,400,30,V,T	385544	385185	- 0.09%	385508	- 0.01%	385508	- 0.01%
	(t)	(277.50)		(277.50)		(1975.34)	
100,400,30,F,L	50496	58773	16.39%	55401	9.71%	51552	2.09%
	(t)	(100.16)		(100.16)		(1300.56)	
100,400,30,F,T	141278	149282	5.67%	146455	3.66%	145144	2.74%
	(t)	(215.71)		(215.71)		(1869.98)	
30,520,100,V,L	53966	56426	4.56%	55358	2.58%	54958	1.84%
	(t)	(995.64)		(995.64)		(3355.96)	
30,520,100,F,L	95294	104117	9.26%	103747	8.87%	99586	4.50%
	(t)	(939.24)		(939.24)		(4032.35)	
30,520,100,V,T	52085	53288	2.31%	52985	1.73%	52985	1.73%
	(t)	(1218.52)		(1218.52)		(3481.08)	
30,520,100,F,T	98357	107894	9.70%	106877	8.66%	105523	7.29%
	(t)	(670.29)		(670.29)		(3927.35)	
30,700,100,V,L	47603	48984	2.90%	48824	2.56%	48398	1.67%
	(1736.05)	(1265.11)		(1265.11)		(4396.42)	
30,700,100,F,L	60525	65356	7.98%	63302	4.59%	62471	3.22%
	(t)	(1479.59)		(1479.59)		(4755.01)	
30,700,100,V,T	45944.5	47083	2.48%	47025	2.35%	47025	2.35%
	(t)	(2426.02)		(2426.02)		(4560.11)	
30,700,100,F,T	55709	58804	5.56%	57886	3.91%	57886	3.91%
	(t)	(1735.72)		(1735.72)		(4866.11)	
30,520,400,V,L	112997.5	125831	11.36%	122048	8.01%	120652	6.77%
	(t)	(5789.27)		(5789.27)		(36530.8)	
30,520,400,F,L	X	177409	-	167837	- 5.40%	161098	- 9.19%
	(t)	(6406.62)		(6406.62)		(429296)	
30,520,400,V,T	X	125518	-	121603	- 3.12%	121588	- 3.13%
	(t)	(6522.23)		(6522.23)		(28214)	
30,520,400,F,T	X	174526	-	171641	- 1.65%	167939	- 3.77%
	(t)	(8415.24)		(8415.24)		(40010.9)	
30,700,400,V,L	X	110000	-	108029	- 1.79%	106777	- 2.93%
	(t)	(12636.2)		(12636.2)		(24816.8)	
30,700,400,F,L	X	165484	-	154215	- 6.81%	148950	- 9.99%
	(t)	(11367.70)		(11367.70)		(69540.1)	
30,700,400,V,T	X	103768	-	102468	- 1.25%	101672	- 2.02%
	(t)	(15879.50)		(15879.50)		(34974.9)	
30,700,400,F,T	X	150919	-	148243	- 1.77%	142778	- 5.39%
	(t)	(11660.40)		(11660.40)		(51877.9)	

dimensions and difficulty of most problem instances treated.

TAB. 3.4 – Distribution of Relative Gaps

P.Set	Card.	X	OPT	IMP	(0-2%]	(2-4%]	(4-6%]	(6-8%]	(8-10%]	>10%
C	43	7	3	2	15	6	4	6	-	-
R	153	-	28	-	46	33	20	11	8	7

Table 3.5 and 3.6 summarize information on the behaviour of the cycle-based tabu search according to various problem characteristics. This information appears under the heading TABU-CYCLE(400). These results are also contrasted to the corresponding behaviour of the path-based tabu search. The statistics are computed over the problem instances of set **R**.

Table 3.5 displays the optimality gap distributions of TABU-PATH and TABU-CYCLE(400) according to the fixed cost and the capacity ratios. One notices, as expected, that more tightly constrained problems are more difficult to solve and thus, for the same computational effort, somewhat less good results are obtained. Compared to the path-based method, however, the cycle-based procedure displays a more consistent behaviour and appears significantly more robust with respect to variation in these two characteristics. Thus, for example, the average gap is not larger than 4.40% contrasted to the almost 20% of TABU-PATH. In fact, the cycle-based methods cuts the optimality gaps in all cases, significantly more so (by at least a factor of 2) as problems become more difficult. Comparing no longer absolute results but rather the general trends displayed by the two approaches, it becomes apparent, however, that the cycle-based method seems to encounter more difficulties when capacities become tighter. A plausible explanation may follow from the fact that when capacities are tight, a large number of moves that consider all flows on arcs will result in infeasible solutions. Then, more time is spent to achieve feasibility and the actual search is less thorough. The superiority of the cycle-based method is not in doubt. Yet, these results point to the need to develop more elaborate search strategies to successfully address the full range of CMND problems. Such a development is beyond the scope of this paper, however.

Table 3.6 displays results for the distribution of optimality gaps according to pro-

TAB. 3.5 – Gap Distribution According to Fixed Cost and Capacity Level

TABU-PATH			
	C1	C2	C8
F01	2.49%	2.21%	2.96%
F05	12.30%	9.16%	6.33%
F10	19.86%	14.45%	8.6%

TABU-CYCLE(400)			
	C1	C2	C8
F01	1.48%	1.31%	1.74%
F05	3.49%	3.69%	4.14%
F10	3.55%	4.27%	4.40%

blem dimensions. The same general behaviour as discussed previously is also observed in this case. For the same computation effort, larger problem instances are, as expected, more difficult to address. On the other hand, this difficulty increases slowly, much slowly than for the path-based procedure. In fact, one observes again the robustness of the cycle-based meta-heuristic with respect to problem dimensions, including the number of commodities. All these results confirm the interest of cycle-based neighbourhoods and the superiority of TABU-CYCLE in addressing CMND problems.

TAB. 3.6 – Gap Distribution According to Problem Dimensions

TABU-PATH						
$\|P\|$	$\|N\|, \|A\|$	opt Gap	$\|N\|, \|A\|$	Opt Gap	$\|N\|, \|A\|$	Opt Gap
10		0.61%		1.81%		2.04%
25	10,25	0.56%	10,50	2.78%	10,75	4.03%
50		1.54%		7.69%		8.96%
TABU-CYCLE(400)						
10		0.00%		0.10%		0.41%
25	10,25	0.78%	10,50	0.48%	10,75	0.91%
50		1.63%		2.47%		2.87%
TABU-PATH						
$\|P\|$	$\|N\|, \|A\|$	Opt Gap	$\|N\|, \|A\|$	Opt Gap	$\|N\|, \|A\|$	Opt Gap
40		5.00%		8.43%		6.47%
100	20,100	9.03%	20,200	16.87%	20,300	20.8%
200		8.06%		24.4%		23.24%
TABU-CYCLE(400)						
40		2.78%		3.50%		2.90%
100	20,100	2.69%	20,200	6.32%	20,300	5.47%
200		5.12%		6.82%		8.19%

To complete our evaluation of the impact of the random seed in the selection of the candidate set, we solved all C problem instances three times. We observed some differences in solution values due to different seeds (no significant differences were

observed in computing times). Differences were relatively small, however, and did not change any of the previous discussions or conclusions. Consequently, to avoid adding to the length of this paper, we do not include these results. The issue is addressed in more details in Ghamlouche, Crainic, and Gendreau (2004).

Summarizing the computational results, after testing on 196 problem instances, we conclude that cycle-based neighbourhoods help build meta-heuristics that yield very good solutions to the CMND. The TABU-CYCLE improves almost all TABU-PATH solutions. The average improvement is around 4.75% while the maximum improvement is 33.7%. Compared to state-of-the-art branch-and-bound methods, the method finds very good quality solutions with a reasonable computation effort. More importantly, cycle-based tabu search displays a robust behaviour relative to variations in problem dimension, including number of commodities, fixed to routing cost ratios, and capacity tightness levels.

3.6 Conclusions

In this paper, we proposed new cycle-based neighbourhood structures for the fixed-charge capacitated multicommodity network design problem. These neighbourhoods are different from traditional ones (add/drop/swap) used in the literature for combinatorial optimization problems. They allow to identify moves based on potential variations in the flow distribution of several commodities and that impact the status of several design arcs simultaneously.

To evaluate the quality of these neighbourhoods, a simple tabu-based local search method has been developed. We realize that this is not a very advanced meta-heuristic. Indeed, it does not include some of the very basic features characteristic of tabu search method. Yet, a simple design does not overshadow the contribution of the proposed neighbourhood structures. We were thus able to clearly analyze their performance and to identify desirable future developments. Numerical experiments have shown this method to display a consistently superior and robust performance. It generates solutions of high quality and outperforms other heuristics proposed in the literature. It is, in fact, the best current heuristic for fixed charge, capacitated, multicommodity network design problems.

Several interesting research avenues are now before us. One avenue consists in studying the impact of the linear programming methodology used to solve the minimum cost network flow problem on the efficiency and quality of the method. This may be particularly important as increasingly larger problem instances are addressed. Parallelization strategies also contribute toward this goal. A second development direction consists in building more complete tabu search meta-heuristics that use cycle-based neighbourhoods. Long-term memory-based strategies (e.g., path relinking and scatter search) will be investigated as efficient means to diversify the search and improve its performance. Preliminary results appear extremely promising (Ghamlouche, Crainic, and Gendreau 2004). Finally, the application of cycle-based neighbourhoods to other difficult combinatorial optimization problems opens up intriguing but challenging perspectives.

References

Ahuja, R.K., Magnanti, T.L., and Orlin, J.B. (1993). *Network Flows – Theory, Algorithms, and Applications.* Prentice-Hall, Englewood Cliffs, NJ.

Balakrishnan, A., Magnanti, T.L., and Mirchandani, P. (1997). Network Design. In Dell'Amico, M., Maffioli, F., and Martello, S., editors, *Annotated Bibliographies in Combinatorial Optimization*, pages 311–334. John Wiley & Sons, New York, NY.

Crainic, T.G., Frangioni, A., and Gendron, B. (2001). Bundle-Based Relaxation Methods for Multicommodity Capacitated Network Design. *Discrete Applied Mathematics*, 112 :73–99.

Crainic, T.G., Gendreau, M., and Farvolden, J.M. (2000). A Simplex-Based Tabu Search Method for Capacitated Network Design. *INFORMS Journal on Computing*, 12(3) :223–236.

Crainic, T.G., Gendreau, M., Soriano, P., and Toulouse, M. (1993). A Tabu Search Procedure for Multicommodity Location/Allocation with Balancing Requirements. *Annals of Operations Research*, 41 :359–383.

Crainic, T.G. and Rousseau, J.-M. (1986). Multicommodity, Multimode Freight Transportation : A General Modeling and Algorithmic Framework for the Service Network Design Problem. *Transportation Research B : Methodology*, 20B :225–242.

Farvolden, J.M. and Powell, W.B. (1994). Subgradient Methods for the Service Network Design Problem. *Transportation Science*, 28(3) :256–272.

Gendron, B., Crainic, T.G., and Frangioni, A. (1998). Multicommodity Capacitated Network Design. In Sansó, B. and Soriano, P., editors, *Telecommunications Network Planning*, pages 1–19. Kluwer Academic Publishers, Norwell, MA.

Gendron, B. and Crainic, T.G. (1994). Relaxations for Multicommodity Network Design Problems. Publication CRT-965, Centre de recherche sur les transports, Université de Montréal, Montréal, QC, Canada.

Gendron, B. and Crainic, T.G. (1996). Bounding Procedures for Multicommodity Capacitated Network Design Problems. Publication CRT-96-06, Centre de recherche sur les transports, Université de Montréal, Montréal, QC, Canada.

Ghamlouche, I., Crainic, T.G., and Gendreau, M. (2004). Path Relinking, Cycle-based Neighbourhoods and Capacitated Multicommodity Network Design. *Annals of Operations Research.* to appear.

Glover, F. and Laguna, M. (1997). *Tabu Search.* Kluwer Academic Publishers, Norwell, MA.

Glover, F. (1986). Future Paths for Integer Programming and Links to Artificial Intelligence. *Computers & Operations Research*, 1(3) :533–549.

Glover, F. (1989). Tabu Search – Part I. *ORSA Journal on Computing*, 1(3) :190–206.

Glover, F. (1990). Tabu Search – Part II. *ORSA Journal on Computing*, 2(1) :4–32.

Holmberg, K. and Yuan, D. (2000). A Lagrangean Heuristic Based Branch-and-Bound Approach for the Capacitated Network Design Problem. *Operations Research*, 48(3) :461–481.

ILOG (1999). *ILOG CPLEX 6.5.* ILOG, Mountain View, CA. U.S.A.

Jarvis, J.J. and Mejia de Martinez, O. (1977). A Sensitivity Analysis of Multicommodity Network Flows. *Transportation Science*, 11(4) :299–306.

Koskosidis, Y.A., Powell, W.B., and Solomon, M.M. (1992). An Optimization–based Heuristic for Vehicle Routing and Scheduling with Soft Time Windows Constraints. *Transportation Science*, 26 :69–85.

Magnanti, T.L. and Wong, R.T. (1984). Network Design and Transportation Planning : Models and Algorithms. *Transportation Science*, 18(1) :1–55.

Minoux, M. (1989). Network Synthesis and Optimum Network Design Problems : Models, Solution Methods and Applications. *Networks*, 19 :313–360.

Powell, W.B. (1986). A Local Improvement Heuristic for the Design of Less-than-Truckload Motor Carrier Networks. *Transportation Science*, 20(4) :246–357.

Appendix

Results for each problem instance of set **R** are given in Tables 4.8 to 4.25. The following information is displayed :

- OPT : The optimal solution obtained by using the branch-and-bound algorithm of CPLEX version 6.5, on one 400MHz processor of a 64-processors Sun Enterprise 10000 with 64 Gigabyte of RAM, operating under Solaris 2.7, and limited to 10 hours CPU time. In this column, a t indicates that the procedure stopped due to the time limit condition.
- TABU-PATH : The value of the solution obtained by the path-based tabu search meta-heuristic on SUN Ultra Sparc1/140 workstations with 64MB of RAM memory, as reported by Crainic, Gendreau, and Farvolden (2000).
- TABU-CYCLE(400) : The value of the solution obtained by the cycle-based tabu search meta-heuristic when performing 400 iterations.
- GAP : The optimality gaps of TABU-PATH and TABU-CYCLE(400) relative to branch-and-bound solutions.
- The figures in parentheses represent total computation time in CPU seconds on the appropriate machines.

TAB. 3.7 – Computational Results R01 : 10,25,10

PROB	OPT	TABU-PATH	GAP	TABU-CYCLE(400)	GAP
R01,F01,C1	74079	74079	0.00%	74079	0.00%
	(0.04)	(5.08)		(8.68)	
R01,F05,C1	92403	92403	0.00%	92403	0.00%
	(0.04)	(5.18)		(8.97)	
R01,F10,C1	115304	115304	0.00%	115304	0.00%
	(0.09)	(4.59)		(8.88)	
R01,F01,C2	84908	84908	0.00%	84908	0.00%
	(0.46)	(8.80)		(9.62)	
R01,F05,C2	113036	114565	1.35%	113036	0.00%
	(1.35)	(8.24)		(11.84)	
R01,F10,C2	147599	151078	2.36%	147599	0.00%
	(2.36)	(8.88)		(11.49)	

TAB. 3.8 – Computational Results R02 : 10,25,25

PROB	OPT	TABU-PATH	GAP	TABU-CYCLE(400)	GAP
R02,F01,C1	232239	232239	0.00%	232239	0.00%
	(0.31)	(18.54)		(19.31)	
R02,F05,C1	322453	326333	1.20%	328005	1.72%
	(1.12)	(16.43)		(25.99)	
R02,F10,C1	419503	424512	1.19%	426866	1.76%
	(1.37)	(11.47)		(26.07)	
R02,F01,C2	316437	316437	0.00%	316437	0.00%
	(0.23)	(29.23)		(27.55)	
R02,F05,C2	431250	431889	0.15%	433442	0.51%
	(0.53)	(29.23)		(34.9)	
R02,F10,C2	559578	564349	0.85%	563570	0.71%
	(0.55)	(26.88)		(29.85)	

TAB. 3.9 – Computational Results R03 :10,25,50

PROB	OPT	TABU-PATH	GAP	TABU-CYCLE(400)	GAP
R03,F01,C1	484830	484830	0.00%	484830	0.00%
	(0.67)	(44.38)		(47.73)	
R03,F05,C1	703362	710840	1.06%	712008	1.23%
	(2.01)	(42.10)		(55.19)	
R03,F10,C1	944990	965330	2.15%	981656	3.88%
	(2.34)	(26.83)		(59.27)	
R03,F01,C2	704247	704247	0.00%	706223	0.28%
	(0.47)	(44.30)		(59.57)	
R03,F05,C2	932897	932897	0.00%	953877	2.25%
	(1.26)	(48.69)		(59.56)	
R03,F10,C2	1188638	1188796	0.01%	1214120	2.14%
	(0.98)	(47.70)		(73.47)	

TAB. 3.10 – Computational Results R04 :10,50,10

PROB	OPT	TABU-PATH	GAP	TABU-CYCLE(400)	GAP
R04,F01,C1	31730	31730	0.00%	31730	0.00%
	(0.02)	(4.10)		(11.51)	
R04,F05,C1	48920	50887	4.02%	48920	0.00%
	(0.06)	(5.39)		(13.36	
R04,F10,C1	63767	67128	5.27%	63767	0.00%
	(0.06)	(5.16)		(12.34)	
R04,F01,C2	33740	33740	0.00%	33740	0.00%
	(0.57)	(8.03)		(13.85)	
R04,F05,C2	53790	53911	0.22%	53790	0.00%
	(2.62)	(6.19)		(14.45)	
R04,F10,C2	74030	75109	1.46%	74030	0.00%
	(1.72)	(6.05)		(14.01)	
R04,F01,C8	68291.7	68310	0.03%	68293	0.00%
	(5.14)	(28.49)		(18.86)	
R04,F05,C8	113004	113283	0.25%	113226	0.20%
	(9.41)	(20.10)		(18.29)	
R04,F10,C8	163208	171481	5.07%	164430	0.75%
	(7.61)	(14.43)		(18.32)	

TAB. 3.11 – Computational Results R05 :10,50,25

PROB	OPT	TABU-PATH	GAP	TABU-CYCLE(400)	GAP
R05,F01,C1	123003	123061	0.05%	123003	0.00%
	(0.38)	(11.85)		(30.41)	
R05,F05,C1	170060	173785	2.19%	170467	0.24%
	(1.51)	(12.84)		(29.98)	
R05,F10,C1	221486	231238	4.40%	221486	0.00%
	(8.44)	(13.47)		(29.4)	
R05,F01,C2	131608	131772	0.12%	131608	0.00%
	(1.58)	(17.37)		(33.54)	
R05,F05,C2	204157	213748	4.70%	205764	0.79%
	(15.60)	(23.23)		(35.08)	
R05,F10,C2	286524	314326	9.70%	292244	2.00%
	(65.12)	(25.91)		(36.49)	
R05,F01,C8	278372	279251	0.32%	278372	0.00%
	(1.45)	(61.25)		(63.25)	
R05,F05,C8	445810	456144	2.32%	449477	0.82%
	(1.99)	(53.78)		(66.36)	
R05,F10,C8	625879	633282	1.18%	629040	0.51%
	(2.20)	(62.49)		(69.07)	

TAB. 3.12 – Computational Results R06 :10,50,50

PROB	OPT	TABU-PATH	GAP	TABU-CYCLE(400)	GAP
R06,F01,C1	245936	248394	1.00%	248615	1.09%
	(3.97)	(47.12)		(56.43)	
R06,F05,C1	401685	426820	6.26%	412283	2.64%
	(51.12)	(47.12)		(66.28)	
R06,F10,C1	559477	676693	20.95%	578752	3.45%
	(205.18)	(35.61)		(66.24)	
R06,F01,C2	286682	289572	1.01%	288460	0.62%
	(25.18)	(78.56)		(69.59)	
R06,F05,C2	498266	534210	7.21%	515967	3.55%
	(463.4)	(47.05)		(80.83)	
R06,F10,C2	734414	846009	15.20%	771683	5.07%
	(1855.94)	(44.00)		(79.68)	
R06,F01,C8	682921	682921	0.00%	683614	0.10%
	(1.47)	(119.29)		(155.22)	
R06,F05,C8	1030479	1042968	1.21%	1051780	2.07%
	(1.45)	(131.228)		(153.05)	
R06,F10,C8	423316	492588	16.36%	438860	3.67%
	(46.56)	(24.5415)		(55.54)	

TAB. 3.13 – Computational Results R07 :10,75,10

PROB	OPT	TABU-PATH	GAP	TABU-CYCLE(400)	GAP
R07,F01,C1	32807	32807	0.00%	32807	0.00%
	(0.08)	(6.24)		(15.65)	
R07,F05,C1	47252	47252	0.00%	47252	0.00%
	(0.13)	(7.30)		(15.3)	
R07,F10,C1	62962	62962	0.00%	62962	0.00%
	(0.62)	(9.31)		(16.25)	
R07,F01,C2	37432	37432	0.00%	37432	0.00%
	(0.66)	(11.35)		(17.04)	
R07,F05,C2	56475	56916	0.78%	56591	0.21%
	(4.39)	(11.20)		(16.85)	
R07,F10,C2	77249	79473	2.88%	78875	2.10%
	(19.01)	(12.17)		(16.99)	
R07,F01,C8	59947	61896	3.25%	59947	0.00%
	(11.05)	(23.01)		(21.84)	
R07,F05,C8	99194	104454	5.30%	100155	0.97%
	(21.75)	(18.94)		(21.57)	
R07,F10,C8	141692	150413	6.15%	142319	0.44%
	(54.39)	(20.40)		(20.61)	

TAB. 3.14 – Computational Results R08 : 10,75,25

PROB	OPT	TABU-PATH	GAP	TABU-CYCLE(400)	GAP
R08,F01,C1	102531	102694	0.16%	102556	0.02%
	(1.33)	(24.87)		(39.13)	
R08,F05,C1	143894	148298	3.06%	143894	0.00%
	(0.54)	(18.64)		(36.77)	
R08,F10,C1	182793	192206	5.15%	182793	0.00%
	(1.35)	(19.77)		(43.04)	
R08,F01,C2	109325	110759	1.31%	109325	0.00%
	(5.45)	(29.33)		(38.43)	
R08,F05,C2	157047	161148	2.61%	158168	0.71%
	(34.23)	(20.56)		(49.01)	
R08,F10,C2	207540	218926	5.49%	208135	0.29%
	(34.23)	(23.06)		(49.54)	
R08,F01,C8	154160	156854	1.75%	155384	0.79%
	(75.38)	(68.92)		(64.34)	
R08,F05,C8	274866.5	296412	7.84%	283133	3.01%
	(404.49)	(39.64)		(49.17)	
R08,F10,C8	415793	452904	8.93%	429896	3.39%
	(776.56)	(54.99)		(74.05)	

TAB. 3.15 – Computational Results R09 : 10,75,50

PROB	OPT	TABU-PATH	GAP	TABU-CYCLE(400)	GAP
R09,F01,C1	171512	172484	0.57%	172343	0.48%
	(2.33)	(37.79)		(73.11)	
R09,F05,C1	296712	313486	5.65%	307038	3.48%
	(32.65)	(32.67)		(76.14)	
R09,F10,C1	424266	546647	28.85%	435590	2.67%
	(218.77)	(30.05)		(84.05)	
R09,F01,C2	192736	195129	1.24%	193242	0.26%
	(10.2)	(77.60)		(89.68)	
R09,F05,C2	357318	400172	11.99%	371998	4.11%
	(143.32)	(61.73)		(104.21)	
R09,F10,C2	522187	644762	23.47%	555945	6.46%
	(1227.52)	(70.58)		(119.60)	
R09,F01,C8	345057	349096	1.17%	348297	0.94%
	(27.83)	(138.66)		(29.67)	
R09,F05,C8	646579	662417	2.45%	669802	3.59%
	(89.5)	662417		(136.86)	
R09,F10,C8	951136	1001559	5.30%	987938	3.87%
	(310.59)	(103.64)		(165.33)	

TAB. 3.16 – Computational Results R10 : 20,100,40

PROB	OPT	TABU-PATH	GAP	TABU-CYCLE(400)	GAP
R10,F01,C1	200087	201350	0.63%	200613	0.26%
	(10.72)	(62.23)		(124.15)	
R10,F05,C1	346813.5	362848	4.62%	350573	1.08%
	(788.2)	(72.47)		(135.11)	
R10,F10,C1	488015	568114	16.41%	507118	3.91%
	(563.29)	(61.78)		(161.92)	
R10,F01,C2	229196	231546	1.03%	232473	1.43%
	(184.31)	(125.36)		(80.32)	
R10,F05,C2	411664	439802	6.84%	432913	5.16%
	(5123.25)	(65.69)		(196.44)	
R10,F10,C2	609104	659189	8.22%	640621	5.17%
	(27260.53)	(95.61)		(175.35)	
R10,F01,C8	486895	489385	0.51%	488737	0.38%
	(391.33)	(704.73)		(464.73)	
R10,F05,C8	951056	993363	4.45%	980010	3.04%
	(555.52)	(421.21)		(414.63)	
R10,F10,C8	1421740	1454329	2.29%	1487270	4.61%
	(608.38)	(299.67)		(332.92)	

TAB. 3.17 – Computational Results R11 : 20,100,100

PROB	OPT	TABU-PATH	GAP	TABU-CYCLE(400)	GAP
R11,F01,C1	714431	726155	1.64%	725416	1.54%
	(392.46)	(423.61)		(392.6)	
R11,F05,C1	1265985	1408514	11.26%	1306090	3.17%
	(t)	(272.10)		(631.13)	
R11,F10,C1	1846295	2392241	29.57%	1914040	3.67%
	(t)	(194.946)		(674.63)	
R11,F01,C2	870451	888165	2.04%	876894	0.74%
	(2960.13)	(661.47)		(580.6)	
R11,F05,C2	1623640	1846121	13.70%	1694860	4.39%
	(21739.5)	(545.29)		(1069.78)	
R11,F10,C2	2414060	2732989	13.21%	2607690	8.02%
	(t)	(429.89)		(1244.41)	
R11,F01,C8	2294912	2308694	0.60%	2295790	0.04%
	(31.51)	(1996.29)		(1126.27)	
R11,F05,C8	3507100	3585266	2.23%	3568430	1.75%
	(79.79)	(1241.06)		(1160.19)	
R11,F10,C8	4579353	4901168	7.03%	4621900	0.93%
	(48.51)	(1262.01)		(786.53)	

TAB. 3.18 – Computational Results R12 : 20,100,200

PROB	OPT	TABU-PATH	GAP	TABU-CYCLE(400)	GAP
R12,F01,C1	1639443	1728210	5.41%	1713670	4.53%
	(4436.74)	(1124.61)		(3012.11)	
R12,F05,C1	3403074.2	4037454	18.64%	3746250	10.08%
	(t)	(903.331)		(3274.92)	
R12,F10,C1	5276171	6415405	21.59%	6070200	15.05%
	(t)	(718.05)		(3612.71)	
R12,F01,C2	2303557	2339682	1.57%	2326230	0.98%
	(1739.79)	(2824.39)		(4700.35)	
R12,F05,C2	4669799	4950073	6.00%	4967940	6.38%
	(3145.59)	(2013.87)		(7932.05)	
R12,F10,C2	7100019	7781907	9.60%	7638050	7.58%
	(1683.84)	(1543.92)		(4728.49)	
R12,F01,C8	7635270	7666421	0.41%	7637250	0.03%
	(78.13)	(3089.45)		(4728.29)	
R12,F05,C8	10067742	1017569	1.07%	10121700	0.54%
	(61.55)	(2870.44)		(2978.28)	
R12,F10,C8	11967768	12958760	8.28%	12079300	0.93%
	(50.29)	(3387.72)		(2284.02)	

TAB. 3.19 – Computational Results R13 : 20,200,40

PROB	OPT	TABU-PATH	GAP	TABU-CYCLE(400)	GAP
R13,F01,C1	142947	143588	0.45%	144138	0.83%
	(6.41)	(153.62)		(277.37)	
R13,F05,C1	263800	284943	8.01%	270316	2.47%
	(547.98)	(174.07)		(274.69)	
R13,F10,C1	365836	439085	20.02%	374999	2.50%
	(116.62)	(174.07)		(259.22)	
R13,F01,C2	150977	152873	1.26%	151513	0.36%
	(90.14)	(215.69)		(252.32)	
R13,F05,C2	282682	308081	8.99%	291510	3.12%
	(3551.86)	(416.29)		(285.3)	
R13,F10,C2	406790	462407	13.67%	420028	3.25%
	(29695.59)	(165.62)		(310.77)	
R13,F01,C8	208088	214345	3.01%	212451	2.10%
	(t)	(416.29)		(393.53)	
R13,F05,C8	446997	491210	9.89%	484112	8.30%
	(t)	(309.69)		(483.27)	
R13,F10,C8	698280	772063	10.57%	758715	8.65%
	(t)	(314.57)		(504.46)	

TAB. 3.20 – Computational Results R14 : 20,200,100

PROB	OPT	TABU-PATH	GAP	TABU-CYCLE(400)	GAP
R14,F01,C1	403414	420837	4.32%	415119	2.90%
	(495.72)	(553.60)		(615.23)	
R14,F05,C1	750091	961760	28.22%	803356	7.10%
	(t)	(518.04)		(822.76)	
R14,F10,C1	1078595	1386426	28.54%	1155840	7.16%
	(t)	(392.21)		(765.56)	
R14,F01,C2	437607	452971	3.51%	453204	3.56%
	(1327.37)	(550.10)		(665.4)	
R14,F05,C2	852330	1003389	17.72%	912456	7.05%
	(t)	(471.52)		(971.27)	
R14,F10,C2	1230749.7	1630263	32.46%	1333440	8.34%
	(t)	(435.05)		(862.72)	
R14,F01,C8	670064	728766	8.76%	702226	4.80%
	(t)	(1688.84)		(1814.58)	
R14,F05,C8	1643376	1873897	14.03%	1748930	6.42%
	(t)	(1688.84)		(3793.69)	
R14,F10,C8	2630332	3006180	14.29%	2882710	9.59%
	(t)	(1455.10)		(3858.42)	

TAB. 3.21 – Computational Results R15 : 20,200,200

PROB	OPT	TABU-PATH	GAP	TABU-CYCLE(400)	GAP
R15,F01,C1	1000787	1087543	8.67%	1049360	4.85%
	(5384.72)	(1209.74)		(1767.13)	
R15,F05,C1	1979413	2912747	47.15%	2158720	9.06%
	(t)	(1025.71)		(2521.43)	
R15,F10,C1	2949264	4729421	60.36%	3135760	6.32%
	(t)	(842.37)		(2618.36)	
R15,F01,C2	1148604	1253329	9.12%	1215130	5.79%
	(t)	(1487.06)		(2794.46)	
R15,F05,C2	2488753	3348931	34.56%	2756680	10.77%
	(t)	(1179.56)		(5980.84)	
R15,F10,C2	3972667	5505598	38.59%	4384640	10.37%
	(t)	(1461.27)		(5133.23)	
R15,F01,C8	2301326.6	2454883	6.67%	2355730	2.36%
	(t)	(7518.55)		(15947.9)	
R15,F05,C8	5573412.8	5839188	4.77%	5926330	6.33%
	(16107.92)	(9170.17)		(13398.5)	
R15,F10,C8	8696932	9547358	9.78%	9180920	5.57%
	(18930.96)	(8497.8)		(12288.05)	

TAB. 3.22 – Computational Results R16 : 20,300,40

PROB	OPT	TABU-PATH	GAP	TABU-CYCLE(400)	GAP
R16,F01,C1	136161	139134	2.18%	136538	0.28%
	(1.05)	(215.20)		(404.89)	
R16,F05,C1	239500	261086	9.01%	247682	3.42%
	(1641.30)	(284.84)		(416.9)	
R16,F10,C1	325671	372377	14.34%	338807	4.03%
	(4345.92)	(236.08)		(418.52)	
R16,F01,C2	138532	139301	0.56%	139973	1.04%
	(27.61)	(309.10)		(376.82)	
R16,F05,C2	241801	257779	6.61%	246014	1.74%
	(851.99)	(309.10)		(430.92)	
R16,F10,C2	337762	371043	9.85%	355610	5.28%
	(8173.47)	(207.44)		(421.58)	
R16,F01,C8	169502	173354	2.27%	172268	1.63%
	(t)	(392.84)		(477.06)	
R16,F05,C8	352976	381866	8.18%	365214	3.47%
	(t)	(298.00)		(520.89)	
R16,F10,C8	541626	570012	5.24%	569874	5.22%
	(t)	(359.34)		(577.86)	

TAB. 3.23 – Computational Results R17 : 20,300,100

PROB	OPT	TABU-PATH	GAP	TABU-CYCLE(400)	GAP
R17,F01,C1	354138	383719	8.35%	370090	4.50%
	(332.10)	(810.79)		(909.17)	
R17,F05,C1	651025	816660	25.44%	688554	5.76%
	(t)	(914.18)		(998.58)	
R17,F10,C1	917956	1323708	44.20%	971151	5.79%
	(t)	(766.44)		(999.38)	
R17,F01,C2	370590	395645	6.76%	380850	2.77%
	(2258.82)	(1199.17)		(927.25)	
R17,F05,C2	708698	865932	22.19%	753188	6.28%
	(t)	(888.50)		(1070)	
R17,F10,C2	1029242	1370038	33.11%	1108180	7.67%
	(t)	(815.94)		(1168.94)	
R17,F01,C8	505310	534490	5.77%	524038	3.71%
	(t)	(2012.43)		(1315.7)	
R17,F05,C8	1112076	1332873	19.85%	1195140	7.47%
	(t)	(1650.51)		(2508.76)	
R17,F10,C8	1847334	2246433	21.60%	1945080	5.29%
	(t)	(1626.07)		(2813.16)	

TAB. 3.24 – Computational Results R18 : 20,300,200

PROB	OPT	TABU-PATH	GAP	TABU-CYCLE(400)	GAP
R18,F01,C1	828559	923006	11.40%	872888	5.35%
	(t)	(1968.72)		(2339.03)	
R18,F05,C1	1542197	2247779	45.75%	1716680	11.31%
	(t)	(2181.04)		(2889.6)	
R18,F10,C1	2292684	3542160	54.50%	2377560	3.70%
	(t)	(2284.64)		(2918.59)	
R18,F01,C2	921762	1017002	10.33%	975396	5.82%
	(t)	(3465.29)		(2677.54)	
R18,F05,C2	1866215	2225384	19.25%	2037950	9.20%
	(t)	(2435.75)		(4335.1)	
R18,F10,C2	2895982	4054910	40.02%	2966370	2.43%
	(t)	(2190.01)		(4234.01)	
R18,F01,C8	1485690.8	1633508	9.95%	1622520	9.21%
	(t)	(6599.88)		(13514.9)	
R18,F05,C8	4010736.7	4454958	11.08%	4576750	14.11%
	(t)	(7160.29)		(28883.1)	
R18,F10,C8	6663506.4	7125415	6.93%	7504310	12.62%
	(t)	(6221.07)		(32695.2)	

Chapitre 4

Path relinking, Cycle-based Neighbourhoods and Capacitated Multicommodity Network Design

La référence de cet article est :
Ghamlouche I., Crainic T.G. and Gendreau M. "Path relinking, Cycle-based Neighbourhoods and Capacitated Multicommodity Network Design", A paraître dans *Annals of Operations Research*

Path relinking, Cycle-based Neighbourhoods and Capacitated Multicommodity Network Design

Abstract

In this paper, we propose a path relinking procedure for the fixed-charge capacitated multicommodity network design problem. Cycle-based neighbourhoods are used both to move along path between elite solutions and to generate the elite candidate set by a tabu-like local search procedure. Several variants of the method are implemented and compared. Extensive computational experiments indicate that the path relinking procedure offers excellent results. It systematically outperforms the cycle-based tabu search method in both solution quality and computational effort and offers the best current meta-heuristic for this difficult class of problems.

Keywords : Path relinking, Tabu search, Fixed-charge capacitated multicommodity network design, Meta-heuristics, Cycle-based neighbourhoods

Résumé

Dans cet article, nous proposons une procédure de "path relinking" pour le problème de synthèse de réseau multiproduits avec capacités. Des mouvements basés sur des cycles sont utilisés autant pour explorer les chemins entre les solutions élites, que pour générer l'ensemble des solutions élites candidates par une procédure de recherche locale avec tabous. Nous développons et comparons plusieurs variantes de la méthode. Les résultats expérimentaux indiquent que la procédure proposée donne d'excellents résultats. Elle offre systématiquement de meilleures performances, en termes de qualité de solution et de temps de résolution que la méthode de recherche avec tabous basée sur des cycles, et offre la meilleure heuristique actuellement disponible pour cette classe de problèmes.

Mots-clés : Chemins reliants, Méthodes de recherche avec tabous, Synthèse de réseau multiproduits avec capacités, Méta-heuristiques, Voisinages par cycles

4.1 Introduction

The fixed-charge capacitated multicommodity network design formulation (CMND) represents a generic model for a wide range of applications in planning the construction, development, improvement, and operations of transportation, logistics, telecommunication, and production systems, as well as in many other major areas (Balakrishnan, Magnanti, and Mirchandani 1997, Magnanti and Wong 1984, Minoux 1989). The problem consists on finding the optimal configuration - the links to include in the final design - of a network of limited capacity to satisfy the demand of transportation of different commodities sharing the network. The objective is to minimize the total system cost, computed as the sum of the link fixed and routing costs. The CMND problem is usually modeled as a combinatorial optimization problem and is NP-hard in the strong sense. Thus, not only the generation of optimal solutions to large problem instances constitutes a significant challenge, but even identifying efficiently good feasible solutions has proved a formidable task not entirely mastered.

A few authors have developed exact solution methods for CMND problems (e.g., Crainic, Frangioni, and Gendron 2001, Gendron, Crainic, and Frangioni 1998, Holmberg and Yuan 2000), but only heuristics have proved to be effective on reasonably large problems with many commodities. Crainic, Gendreau, and Farvolden (2000) proposed a tabu search meta-heuristic for the path-based formulation of the problem. The method explores the space of the path-flow variables by using pivot-like moves and column generation. Even though the method produced very impressive results, its search efficiency may be limited by the fact that each move considers the impact of changing the flow of one commodity only (a pivot from one path to another). Moreover, one cannot guarantee that all paths will be considered, neither explicitly, nor implicitly, which impairs the capabilities of the algorithm to explore thoroughly the search space.

Ghamlouche, Crainic, and Gendreau (2003) proposed a new class of neighbourhood structures for the CMND that overcomes some of these limitations. To evaluate these neighbourhoods, the authors implemented them into a very simple tabu-based local search procedure that currently appears as the best approximate solution method for the CMND in terms of robust performance, solution quality, and computing

efficiency. This approach is not entirely satisfactory, however, since it does not go beyond a rather local exploration of the search space. In fact, Ghamlouche, Crainic, and Gendreau (2003) clearly state that their goal was to explore the behaviour of the neighbourhood class only, and that more refined search methods would have to be developed. The present paper is a contribution toward reaching this goal.

Path relinking has been proposed as a method to better explore the solution space of complex problems when a set of promising, or *elite* solutions is known (Glover 1997, Glover and Laguna 1997, Glover, Laguna, and Marti 2000). The fundamental idea of the method is to build a solution path between two elite solutions, using their attributes to guide the generation of intermediary solutions. This process is supposed to yield solutions that combine the "good" characteristics of the starting solutions and that improve the set of elite solutions and, eventually, the overall best. The body of research reported on applications of path relinking is somewhat limited but appears to sustain these expectations.

The motivation of the research presented in this paper is to explore the adaptation of path relinking to the CMND. We aim, in particular, to evaluate the benefits of combining the path relinking framework and the cycle-based neighbourhood structures of Ghamlouche, Crainic, and Gendreau (2003) into a better meta-heuristic for the CMND. Numerical experiments support this claim and show that combining cycle-based tabu search and path relinking results in an enhanced and more effective search strategy.

To summarize, the contributions of this work are : 1) a new meta-heuristic for the CMND that outperforms the current best approximate methods; 2) a successful development of a path relinking method to a difficult combinatorial optimization problem; 3) an enhancement of our understanding of the behaviour of path relinking as a search method.

The outline of the paper is as follows. Section 4.2 starts with a formal definition of the problem and summarizes the relevant details of the cycle-based neighbourhoods and initial tabu search method. Following a brief outline of the path relinking methodology, Section 4.3 details our implementation for the CMND. Section 4.4 is dedicated to experimental results. We compare, in particular, the path relinking results to those

obtained by the cycle-based tabu search to identify the effects of the new methodology on the quality of the solutions obtained and total computational time. We conclude in Section 4.5.

4.2 Background and Neighbourhood Overview

Let $\mathcal{G} = (\mathcal{N}, \mathcal{A})$ be a network with set of nodes $\mathcal{N}$ and set of directed arcs $\mathcal{A}$. Let $\mathcal{P}$ denote the set of commodities to move using this network. For each $p \in P$, let w^p denote the required amount of flow of commodity p to be shipped from its origin $o(p)$ to its destination $s(p)$. The total flow on each arc $(i,j) \in \mathcal{A}$ is limited by the capacity u_{ij}. There are two costs involved in the network. The unit cost of moving commodity $p \in P$ through the arc (i,j), denoted c_{ij}^p, and the fixed cost of including arc (i,j) in the design of the network, denoted f_{ij}. The problem consists in minimizing the sum of all costs while satisfying the demand of transportation. The arc-based formulation of the CMND can then be written as

$$\min z(x,y) = \sum_{(i,j)\in\mathcal{A}} f_{ij}y_{ij} + \sum_{p\in\mathcal{P}}\sum_{(i,j)\in\mathcal{A}} c_{ij}^p x_{ij}^p \tag{4.1}$$

subject to

$$\sum_{j\in\mathcal{N}^+(i)} x_{ij}^p - \sum_{j\in\mathcal{N}^-(i)} x_{ji}^p = \begin{cases} w^p & if\ i = o(p) \\ -w^p & if\ i = s(p) \\ 0 & \text{otherwise.} \end{cases} \quad \forall i \in \mathcal{N}, \forall\ p \in \mathcal{P}, \tag{4.2}$$

$$\sum_{p\in\mathcal{P}} x_{ij}^p \leq u_{ij}y_{ij} \quad \forall (i,j) \in \mathcal{A}, \tag{4.3}$$

$$x_{ij}^p \geq 0 \quad \forall (i,j) \in \mathcal{A}, \forall p \in \mathcal{P}, \tag{4.4}$$

$$y_{ij} \in \{0,1\} \quad \forall (i,j) \in \mathcal{A} \tag{4.5}$$

where y_{ij}, $(i,j) \in \mathcal{A}$, represent the design variables that equal 1 if arc (i,j) is selected in the final design (and 0 otherwise), x_{ij}^p stand for the flow distribution decision variables indicating the amount of flow of commodity $p \in \mathcal{P}$ on arc (i,j), and $\mathcal{N}^+(i)/\mathcal{N}^-(i)$ denote the set of outward/inward neighbours of node i.

The objective function (4.1) accounts for the total system cost, the fixed cost of arcs included in a given design plus the cost of routing the product demand, and aims to select the minimum cost design. Constraints (4.2) represent the network flow conservation relations, while constraints (4.3) state that for each arc, the total flow of all commodities cannot exceed its capacity if the arc is opened ($y_{ij} = 1$) and must be 0 if the arc is closed ($y_{ij} = 0$). Relations (4.5) and (4.4) are the usual non-negativity and integrality constraints for decision variables.

Recall that, for a given design vector $\bar{y}$, the arc-based formulation of the CMND becomes a capacitated multicommodity minimum cost flow problem (CMCF)

$$\min z(x(\bar{y})) = \sum_{p \in \mathcal{P}} \sum_{(i,j) \in \mathcal{A}(\bar{y})} c_{ij}^p x_{ij}^p \tag{4.6}$$

subject to (4.2) plus

$$\begin{aligned} \sum_{p \in \mathcal{P}} x_{ij}^p &\leq u_{ij} \bar{y}_{ij} \quad \forall (i,j) \in \mathcal{A}(\bar{y}), \\ x_{ij}^p &\geq 0 \quad \forall (i,j) \in \mathcal{A}(\bar{y}), \forall p \in \mathcal{P}, \end{aligned}$$

where $\mathcal{A}(\bar{y})$ stands for the set of arcs corresponding to the design $\bar{y}$. A solution to the CMND may then be viewed as an assignment $\bar{y}$ of 0 or 1 to each design variable, plus the optimal flow of the corresponding multicommodity minimum cost flow problem $x^*(\bar{y})$. Similarly, the objective function value associated to a solution $(\bar{y}, x^*(\bar{y}))$ is the sum of the fixed cost of the open arcs in $\bar{y}$ and the objective function value of the CMCF associated to $x^*(\bar{y})$

$$z(\bar{y}, x^*(\bar{y})) = \sum_{(i,j) \in \mathcal{A}(\bar{y})} f_{ij} \bar{y}_{ij} + z(x^*(\bar{y})). \tag{4.7}$$

The cycle-based neighbourhood structures for the CMND proposed by Ghamlouche, Crainic, and Gendreau (2003) explore the space of the arc design variables by re-directing flow around cycles and closing and opening design arcs accordingly. The basic neighbourhood defines moves that may explicitly take into account the impact on the total design cost of potential modifications to the flow distribution of several

commodities simultaneously. The fundamental idea is that one may move from one solution to another by 1) identifying two points in the network together with two paths connecting these points, thus closing a cycle; 2) deviating the total flow from one path to another such that at least one currently open arc becomes empty; 3) closing all previously open arcs in the cycle that are empty following the flow deviation and, symmetrically, opening all previously closed arcs that now have flow. Such neighbourhoods are huge, however, and their explicit and exhaustive exploration is not practical in most situations. Moreover, the complete evaluation of any design modification involves the resolution of a capacitated multicommodity network flow problem, which rapidly becomes extremely computation intensive. Thus, in order to select the best move for a given solution, the method implements an efficient procedure that avoids the complete evaluation of every examined move and generates a limited number of cycles that include the "good" moves.

Indeed, not all cycles are of equal interest. The method seeks, in particular, moves that modify the status of several arcs and that lead to a significant modification of the flow distribution. Therefore, moves that close at least one arc and open new paths for a group of commodities appear attractive. To close an arc (i, j), one must be able to deviate all its flow. Let the *residual capacity* of a cycle denote the maximum flow one can deviate around the cycle. The residual capacity of any cycle that includes (i, j) must then be at least equal to $\sum_{p \in \mathcal{P}} x_{ij}^p$, the total flow on arc (i, j). Consequently, the cycles of interest are those that display a residual capacity equal to one of the values in Γ defined as the set of the total (strictly positive) volumes on the open arcs of the corresponding network :

$$\Gamma(\bar{y}) = \left\{ \sum_{p \in \mathcal{P}} x_{ij}^p > 0 \ : \ (i, j) \in \mathcal{A}(\bar{y}) \right\}. \tag{4.8}$$

Cycles are thus identified on *residual networks* of $\mathcal{A}(\bar{y})$ defined according to $\Gamma(\bar{y})$. The best move in the neighbourhood of $\bar{y}$ is then identified by the cycle leading to the network modification that yields the largest improvement (smallest deterioration) in the objective function (4.7). To reduce the computational burden, cycles are identified and evaluated for a set of candidate links. The "lowest" cost cycle for

1 : Build set $\Gamma(\bar{y})$
2 : For each $\gamma \in \Gamma(\bar{y})$
    Build the γ-residual network
    For each arc $(i,j) \in \mathcal{C}(\gamma)$
        find the lowest cost cycle using a network optimization procedure
3 : Select and implement the best move
4 : Evaluate new solution

FIG. 4.1 – Neighbourhood Exploration

each candidate link is identified by an optimization heuristic based on a modification of the shortest path label-correcting algorithm that avoids getting trapped in negative directed cycles. The method thus progressively builds a set of good candidate neighbours (cycles) among which the best move is selected.

Let $\mathcal{C} \subseteq \mathcal{A}$ denote the set of *candidate* links, where each arc $(i,j) \in \mathcal{C}$ will be considered as the starting point of cycles. Let $\mathcal{C}(\gamma) \subseteq \mathcal{C}$ include all arcs $(i,j) \in \mathcal{C}$ that can support a movement of γ units of flow. A closed arc may belong to $\mathcal{C}(\gamma)$ only if its capacity is at least γ. For an open arc, either its flow or its residual capacity must be at least γ units. The heuristic procedure proposed to explore the neighbourhoods of a given solution $\bar{y}$, given a set of candidate links $\mathcal{C}(\bar{y})$, is synthesized in Figure 4.1.

As explained in Ghamlouche, Crainic, and Gendreau (2003), the *γ-residual network* corresponding to a flow variation of γ units is built by replacing each arc (i,j) of the original network by at most two arcs $(i,j)^+$ and $(j,i)^-$. Arc $(i,j)^+$ is included only if an additional amount of γ units of flow may pass on arc (i,j), that is if its residual capacity $u_{ij} - \sum_{p \in \mathcal{P}} x_{ij}^p$ is greater or equal to γ. The cost c_{ij}^+ associated to $(i,j)^+$ approximates the cost of routing γ units of additional flow on arc (i,j). It equals the average commodity routing costs on the arc, plus the fixed cost if it is currently closed :

$$c_{ij}^+ = \frac{\sum_{p \in \mathcal{P}} c_{ij}^p}{\mid P \mid} \gamma + f_{ij}(1 - (\lceil \min(1, \sum_{p \in \mathcal{P}} x_{ij}^p) \rceil)).$$

Symmetrically, arc $(j,i)^-$ is not included in the γ-residual network if the total flow on arc (i,j), $\sum_{p \in \mathcal{P}} x_{ij}^p$ is less than γ. Otherwise, we associate to $(j,i)^-$ the cost c_{ji}^- that

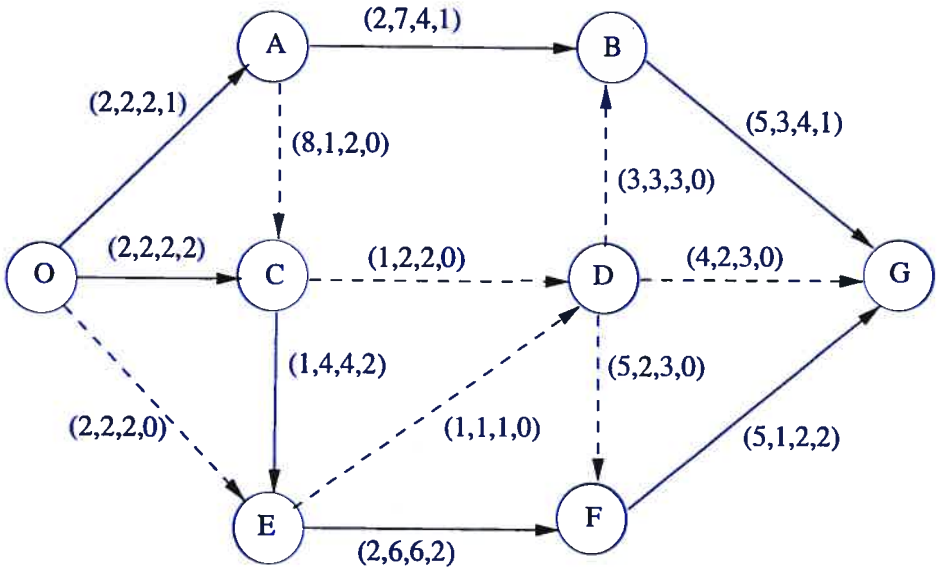
approximates the value of a reduction of γ units of the total flow (all commodities) currently using arc (i, j). This approximation is computed as the weighted average of the routing costs of the commodities currently using arc (i, j). The fixed cost of (i, j) is then subtracted if the reduction of γ units of flow leaves the arc empty :

$$c_{ji}^{-} = \begin{cases} -\frac{\sum_{p \in \mathcal{P}} c_{ij}^{p} x_{ij}^{p}}{\sum_{p \in \mathcal{P}} x_{ij}^{p}} \gamma - f_{ij} & \text{if } \sum_{p \in \mathcal{P}} x_{ij}^{p} = \gamma \\ -\frac{\sum_{p \in \mathcal{P}} c_{ij}^{p} x_{ij}^{p}}{\sum_{p \in \mathcal{P}} x_{ij}^{p}} \gamma & \text{if } \sum_{p \in \mathcal{P}} x_{ij}^{p} > \gamma \end{cases}$$

To illustrate the construction of residual networks in our context, consider the example of Figures 4.2 and 4.3 (Ghamlouche, Crainic, and Gendreau 2003), where a 2-residual network is built. Four attributes are indicated on arcs : fixed cost, routing cost (only one commodity is considered), capacity, and total flow. One unit of flow passes on arc (O, A) and its residual capacity is equal to 1. Consequently, one can neither send nor take out 2 units of flow and, thus, there are no arcs between O and A in Figure 4.3. Arc (O, C) is saturated, thus no more flow can be routed from O to C and the associated arc does not appear in Figure 4.3. On the other hand, reducing the flow of arc (O, C) by 2 units of flow leaves the arc empty, and thus the (C, O) arc is included in the 2-residual network with a cost -6 representing the savings in routing and fixed costs. The rest of the arcs are treated similarly.

To evaluate the concepts introduced, Ghamlouche, Crainic and Gendreau (2003) developed a simple tabu search-based local search procedure that integrates two versions of the cycle-based neighbourhood. The basic one corresponds to the definition above and considers the flow of all commodities when determining cycles. The second one refines the search by implementing moves that result from the deviation of the flow of one commodity at a time (the set Γ is instantiated for each commodity, denoted Γ^p, to contain the flow of the particular commodity that exists on each arc of the network), and is used to intensify the search.

Following an initialization phase, the procedure explores the search space defined by the design variables using a simple local search framework with tabu tags. At each iteration, the best non-tabu move is determined and implemented, whether it improves the overall solution or not. A short-term tabu memory is used to record cha-



(fixed cost, routing cost, capacity, flow)

FIG. 4.2 – Network Example

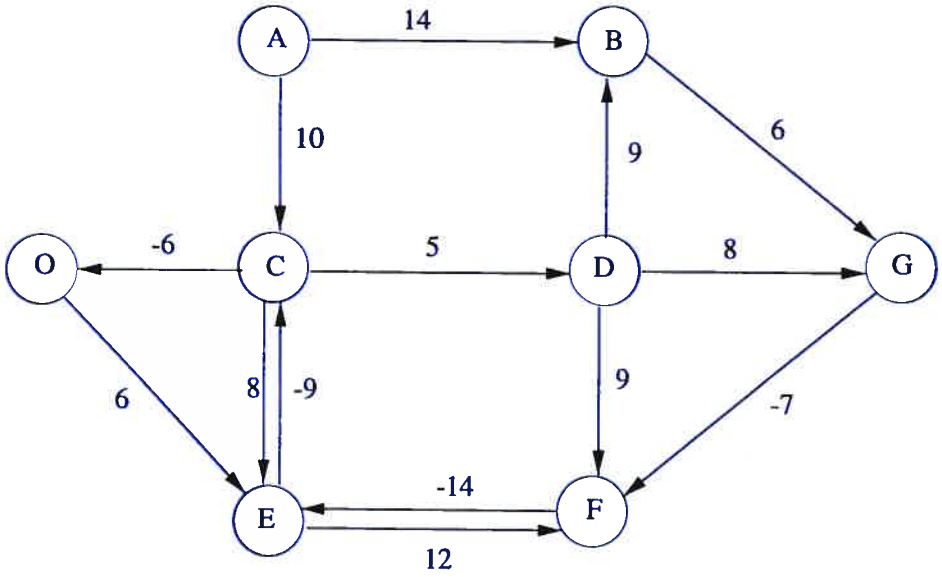


FIG. 4.3 – Associated 2-Residual Network

racteristics of visited solutions to avoid cycling. When a particularly good solution is encountered, the search is intensified using the particular neighbourhood implementation indicated above, which considers the flow distribution of one commodity only. A solution is considered particularly good when it either improves the best overall solution or is close to it by at least a pre-defined percentage. The method terminates whenever a predefined stopping criterion (number of iterations, CPU time, etc.) is met. Computational results on a large set of instances with various characteristics, show that the cycle-based tabu search is powerful. It is in fact the best current heuristic for the CMND.

The cycle-based tabu search has been extensively calibrated and tested. This experimental study yielded the parameter settings that are also used to implement the path relinking presented in the following section.

4.3 Path Relinking

Introduced by Glover (1997; see also Glover and Laguna 1997 and Glover, Laguna, and Marti 2000), *path relinking* is a meta-heuristic that operates on a set of elite solutions, called the *reference set*, and generates paths between solutions in this set to create improved new ones. Starting from an *initial* solution, the primary goal of the search is to find a path towards another, *guiding* solution, by performing moves that progressively introduce into the current solution attributes of the guiding solution. Thus, the method does not progress by choosing a "best" move from the neighbourhood, but rather by selecting the "best" move from a restricted set of moves that incorporate some or all of the "good" attributes of the guiding solution. This exploration allows the search to perform moves that may be considered unattractive according to the objective function value but which appear essential in reaching solutions with given characteristics.

Path relinking has been proposed in connection with tabu search. Very little computational investigation of this method has been done, yet. Laguna and Marti (1999) apply path relinking within a GRASP framework to the problem of minimizing the number of arc crossings in a bipartite graph. Results are very encouraging : the authors report that path relinking improved the performance of the GRASP implementation.

1 : Generate a starting set of solutions using the cycle-based tabu search
2 : Designate a subset of solutions to be included in the reference set
While the reference set cardinality > 1
- Select two solutions from the reference set
- Identify initial and guiding solutions
- Remove initial solution from the reference set
- Move from initial toward guiding solution, generating intermediate solutions
- Update the reference set

3 : Verify stopping criterion : Stop or go to 1 :

FIG. 4.4 – General Path Relinking Procedure

Laguna, Marti, and Campos (1999) propose an algorithm based on tabu search with strategies for search intensification and diversification for the linear ordering problem. The method uses path relinking for search intensification. Elite solutions are selected as the best solutions found during the entire search, while initial solutions are local optima. The authors report that combining tabu search and path relinking is beneficial. They also indicate, however, that one must find the "right" balance between search intensification and diversification in order to obtain the best results. Ribeiro, Uchoa, and Werneck (2002) propose a hybrid GRASP with perturbations and path relinking for the steiner problem in graphs. Their results indicate that path relinking is effective and systematically leads to improvements in the solutions found at the end of the hybrid GRASP perturbation.

This paper investigates the implementation of path relinking to the CMND. In our implementation, the method starts with a tabu search phase using the algorithm of Ghamlouche, Crainic, and Gendreau (2003) to build the reference set. When a predefined number of consecutive moves without improvement is observed, the method switches to a path relinking phase that stops when the reference set becomes empty (cardinality ≤ 1). Then, either stopping conditions are verified, or the procedure is repeated to build a new reference set. The procedure is schematically shown in Figure 4.4.

The path relinking method is made up of the following three main components :

- Identification of the reference set.
- Choice of the initial and guiding solutions.
- Identification of the neighbourhood structure and the guiding attributes.

The following sub-sections detail our implementation of these components to address the CMND. Section 4.3.4 sums up the overall path relinking procedure.

4.3.1 Identification of the Reference Set

What solutions are included in the reference set $\mathcal{R}$, how good and how diversified they are, have a major impact on the quality of the new generated solutions. In our implementation, $\mathcal{R}$ is built during the tabu search phase and subsequently enriched during the path relinking phase. We study six strategies corresponding to different ways to build $\mathcal{R}$.

In strategy *S1*, $\mathcal{R}$ is built using each solution that, at some stage of the tabu search phase, improves the best overall solution and become the best one. The goal of this strategy is to link *overall improving* solutions.

Strategy *S2* retains the *"best" local minima* found during the tabu search phase. This strategy is motivated by the idea that local minimum solutions share characteristics with optimum solutions. One then hopes that linking such solutions yields improved new ones. In Figure 4.5, solutions 2, 3, 4, and 5 are members of $\mathcal{R}$ when its dimension is limited to 4.

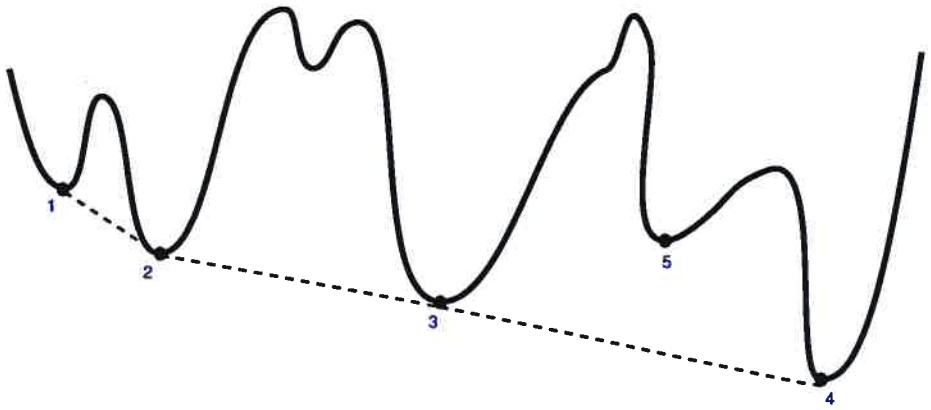


FIG. 4.5 – Building the Reference Set Example

Strategy *S3* selects $\mathcal{R}$-*improving* local minima, that is local minimum solutions that offer a better evaluation of the objective function than those already in $\mathcal{R}$. The idea is to introduce a time dimension in the selection process - often better

For a given candidate solution s

If $\mathcal{R}$ is not full, Add s to $\mathcal{R}$

Else, If s is the best overall solution,
Replace the worst solution in $\mathcal{R}$ by s

Else, If s is better than the worst solution in $\mathcal{R}$

- Compute $Median$
- Compute D_s^b
- If $D_s^b > Median$,
  Replace the worst solution in $\mathcal{R}$ by s

FIG. 4.6 – Building $\mathcal{R}$ According to Strategy *S4*

solutions are encountered once the search has proceeded for some time - without loosing potentially good solutions found early on during the search. In Figure 4.5, if solutions 1 and 2 are already in $\mathcal{R}$, only solutions 3 and 4 are admitted to this set.

Strategy *S4* represents a first attempt to allow solutions to be retained in $\mathcal{R}$ not only according to an attractive solution value but also according to a *diversity*, or *dissimilarity* criterion. Define D_s^b, the level of dissimilarity between solution s and the best solution b, as the number of arcs with different status between the two solutions :

$$D_s^b = \sum_{(i,j)\in A} d_{ij},$$

where

$$d_{ij} = \begin{cases} 0 & if \quad y_{ij}^s = y_{ij}^b \\ 1 & \text{otherwise.} \end{cases}$$

Define also the median position of all solutions $s_i \in \mathcal{R}$ relatively to the best solution b as :

$$Median = \frac{\sum_{s_i \in \mathcal{R}}^{s_i \neq b} D_{s_i}^b}{|\mathcal{R}| - 1},$$

where $|\mathcal{R}|$ denotes the dimension of the reference set. A solution s is then included in $\mathcal{R}$ if it improves the best overall solution, or if it improves the worst solution and its level of dissimilarity exceeds the median, $D_s^b > Median$. The procedure of Figure 4.6 is used to implement this strategy.

Strategy *S5* aims to ensure both the *quality* and the *diversity* of solutions in $\mathcal{R}$. This strategy is pointed out by Laguna and Armentano (2001) as the most important

lesson they learned related to the way the reference set should be initialized. Starting with a large set of ("good") solutions S, $\mathcal{R}$ is partially filled with the best solutions found, to satisfy the purpose of quality. It is then extended with solutions that change significantly the structure of the solutions already in $\mathcal{R}$ to ensure diversity. Strategy *S5* thus builds the reference set according to the following steps :

1. Initialize the reference set $\mathcal{R}$ with solutions satisfying strategy *S1*.
2. For each solution $s \in \{S \backslash \mathcal{R}\}$, compute the level of diversity Δ_s^R between solution s and all solutions $s_i \in \mathcal{R}$: $\Delta_s^R = \sum_{s_i \in \mathcal{R}} D_{s_i}^s / |\mathcal{R}|$
3. Extend $\mathcal{R}$ with solutions $s \in \{S \backslash \mathcal{R}\}$ that maximize Δ_s^R.

Strategy *S6* proceeds similarly to *S5* with the difference that $\mathcal{R}$ is extended with solutions *close* to those already in $\mathcal{R}$. This strategy aims to *intensify* the search by grouping good solutions with similar designs. To build $\mathcal{R}$, strategy *S6* implements the first two steps used for strategy *S5*, then extends it with solutions $s \in \{S \backslash \mathcal{R}\}$ that minimize Δ_s^R.

4.3.2 Choice of the Initial and Guiding Solutions

The choice of the initial and guiding solutions is also critical to the quality of the new solutions and, thus, the performance of the procedure. We investigate the effect of the following criteria :

C1 : Guiding and initial solutions are defined as the *best* and *worst* solutions, respectively.

C2 : Guiding solution is defined as the *best* solution in the reference set, while the initial solution is the *second best* one.

C3 : Guiding solution is defined as the *best* solution in the reference set, while the initial solution is defined as the solution with *maximum Hamming distance* from the guiding solution.

C4 : Guiding and initial solutions are chosen *randomly* from the reference set.

C5 : Guiding and initial solutions are chosen as the *most distant* solutions in the reference set.

C6 : Guiding and initial solutions are defined respectively as the *worst* and the *best* solutions in the reference set.

4.3.3 Neighbourhood Structure and Guiding Attributes

Recall (Section 4.2) the two versions of the cycle-based neighbourhood used in the tabu search procedure : the basic one that considers all flows on arcs, while the intensification version deviates only a commodity at the time. The former brings about significant modifications to the design structure, while the latter yields more limited changes. The aim of the path relinking procedure is to introduce progressively attributes of the guiding solution into new solutions obtained by moving away from an initial point. This requires small but purposeful movements and, thus, we use the second variant of the neighbourhood structure considering flow deviations of only one commodity at a time. Consequently, for the path relinking phase, the set Γ becomes Γ^p, defined as the set of the positive flow of commodity p on each of the open arcs of the corresponding network :

$$\Gamma^p(\bar{y}) = \left\{ x_{ij}^p > 0 \ : \ (i,j) \in \mathcal{A}(\bar{y}),\ p \in \mathcal{P} \right\}. \tag{4.9}$$

Moves from the current solution to a neighbouring one during path relinking must direct the search towards the guiding solution. Thus, cycles of interest are those that introduce attributes (i.e, arcs) present in the guiding solution. To apply this aggressive orientation judiciously, the set of the candidate links $\mathcal{C}$ becomes $\mathcal{C}_r$, the set of all arcs with a different status in the initial solution s_i and the guiding solution s_g :

$$\mathcal{C}_r = \{(i,j) \in \mathcal{A} : y_{ij}^{s_i} \neq y_{ij}^{s_g}\}. \tag{4.10}$$

Similarly, sets $\mathcal{C}_r(\gamma) \subseteq \mathcal{C}_r$ include now all arcs $(i,j) \in \mathcal{C}_r$ that can support a movement of γ units of flow. To select among the various possible paths between the two reference solutions, we move from the initial solution toward the guiding solution by selecting at each step the "best" move with respect to the improvement in the objective function. That is, cycles that include arcs in $\mathcal{C}_r$ are evaluated and a move is performed toward the "best" one. This neighbour becomes the new current solution, even if it is worse than the previous one.

Initialization

Generate an initial feasible solution to initiate *BestSolution* and *CurrentSolution*

Main search loop

Repeat the following until a stopping condition is met

- Perform one tabu search iteration to get a new current solution
- Add current solution to $\mathcal{R}$ if it satisfies the requirements of strategy S_i
- If path relinking is to be launched,
  - Extend $\mathcal{R}$ if strategy *S5* or *S6* is used
  - Perform *Main Path Relinking Loop* as shown in Figure 4.8

FIG. 4.7 – Path Relinking Procedure

To select the "best" move in the neighbourhood of the current solution $\bar{y}$ during the path relinking phase, the procedure first determines the set of flow deviations $\Gamma^p(\bar{y})$ and the set of candidate links C_r as defined in (4.9) and (4.10) respectively. Then, for each commodity $p \in P$, a γ-residual network is built for each flow value $\gamma \in \Gamma^p$ and a low cost cycle is found for each arc $(i,j) \in C_r(\gamma)$. If the lowest-cost cycle for commodity p is negative, the procedure updates the current solution, assigns a tabu status to each complemented arc, and examines the next product. If no negative cycle is found for any commodity, the procedure selects the cycle with lowest-cost overall cost and implements it. A new relinking path is initiated when no cycle is found between the current and guiding solutions.

4.3.4 The Overall Procedure

To sum up, the procedure combines cycle-based tabu search and path relinking and alternates between the two. After an initialization phase, the search proceeds with the tabu search to initialize the reference set $\mathcal{R}$ according to one of the strategies of Section 4.3.1. The method then switches to a path relinking phase. Initial and guiding solutions are selected in $\mathcal{R}$ according to one of the strategies of Section 4.3.2. The search proceeds by moving towards the guiding solution as indicated in Section 4.3.3 and Figure 4.8.

Notice that, due to the nature of the neighbourhoods used, there is no guarantee that the guiding solution will be reached. One cannot, therefore, stop the process only if the current and the guiding solutions are the same. We then define Δ_{IG} as the

1 : While the reference set cardinality > 1
- Identify *InitialSolution* and *GuidingSolution* according to a strategy C_i
- Set *CurrentSolution* to *InitialSolution* and remove the latter from $\mathcal{R}$
- Compute Δ_{IG}

2 : While ($\Delta_{IG} \neq 0$ AND *CurrentSolution* $\neq$ *GuidingSolution*)
- Set *BestPositiveCycle* $= \infty$
- Repeat until no negative cycle is detected in any γ-residual network
  - ► For each commodity $p \in P$ and flow value $\gamma \in \Gamma^p$
    - Build the γ-residual network
    - Find a low cost cycle for each arc $(i,j) \in \mathcal{C}_r(\gamma)$, $\gamma \in \Gamma^p$
    - Select the best overall cycle, γ_{best} the associated volume of deviated flow
    - If the selected cycle is improving (negative cost) update the solution by :
      - Modify the flow around the selected cycle by a quantity γ_{best} ;
      - Open/close appropriate arcs ;
      - Update the cost of *CurrentSolution* by adding the cost of the cycle (without solving the CMCF).
      - Assign a tabu status to each complemented arc.
    - Otherwise, update *BestPositiveCycle* (positive cost cycle)
- If no improving cycle was found
  - ► If *BestPositiveCycle* $< \infty$, implement corresponding move
    - Open/close appropriate arcs
    - Assign a tabu status to each complemented arc
  - ► Otherwise, go to 1 (null cycle found)
- Solve exactly the CMCF associated to the current design to obtain the new *CurrentSolution*
- $\Delta_{IG} = \Delta_{IG} - 1$
- Update *BestSolution* (i.e., cost(*CurrentSolution*) $<$ cost(*BestSolution*))
- Update $\mathcal{R}$ if cost(*BestSolution*) <cost(Best of $\mathcal{R}$) and cost(*CurrentSolution*) $>$ cost(*BestSolution*)
  - Perform an intensification phase on *BestSolution*
  - Add *BestSolution* to $\mathcal{R}$ and go to 1

FIG. 4.8 – Main Path Relinking Loop

number of arcs with different status between the initial and the guiding solutions and we allow the search to explore a number of solutions not larger than Δ_{IG}.

Computing the solution of the capacitated multicommodity network flow subproblem constitutes a significant computational effort. On the other hand, since modifying the flow of one commodity around a cycle does not, in most cases, induce large variations in the design of the network, the CMCF is recomputed only when no more negative-cost cycles are found. To further reduce the computational burden, the reference set $\mathcal{R}$ is updated only with the best solutions found during a path relinking exploration, and only if these solutions improve the best overall solution (always included in $\mathcal{R}$).

It is clear that the path relinking approach induces a trajectory not easily accomplished by the initial tabu search procedure. This claim is supported by the results reported in the next section, which also tend to show that improved solutions are found by using the path relinking procedure we propose.

4.4 Experimental results

The objectives of the experiments is to calibrate and analyze the proposed path relinking procedure.

Other than identifying appropriate values for a few key parameters (Section 4.4.1), the calibration phase aims to explore the impact on the solution quality of each combination of strategies to build the reference set (Section 4.3.1) and select initial and guiding solutions (Section 4.3.2). We present these results in Section 4.4.2, while the influence of randomness is examined in Section 4.4.3. Experimental analyses as well as comparisons with the cycle-based tabu search and branch-and-bound are provided in Section 4.4.4.

The computer code is written in C++. The exact evaluation of the capacitated multicommodity network flow problems is done using the linear programming solver of CPLEX 6.5 (1999). All tests were conducted on one 400MHz processor of a 64-processors Sun Enterprise 10000 with 64 Gigabyte of RAM, operating under Solaris 2.7. Computing times are reported in CPU seconds.

4.4.1 Parameter settings

Two parameters have to be calibrated : the cardinality of the reference set $|\mathcal{R}|$ and the criteria to launch a path relinking phase. Three values, 6, 8, and 15, were tested for $|\mathcal{R}|$. The path relinking phase is started after a number of iterations of the tabu search procedure without improvement of the best value of the objective function. We retained 10, 20, and 50 iterations for this parameter, $PRstart$. Thus, in total, 9 sets of parameters were tested.

Strategies *S3* and *C1*, for building the reference set and selecting initial and guiding solutions, respectively, were used for these tests. Experiments were conducted on 10 problems covering networks from 100 to 700 design arcs and from 10 to 400 commodities. For more details on these problems, refer to Ghamlouche, Crainic, and Gendreau (2003).

For each parameter set, we run the path relinking algorithm 3 times on each problem. The average gap for each problem was calculated as the average of the gaps between the path relinking solutions and the best known solution for the same problem (that of the branch-and-bound procedure of CPLEX 6.5, when available, or that obtained by Ghamlouche, Crainic, and Gendreau 2003, otherwise). The average gap assigned for each parameter set is the average of the average gaps of all problems. Moreover, we assign to each problem a score from 9 to 1 for each parameter set. This score depends on the value of the average gap and is determined as follows : for each problem, a score of 9 is assigned to the parameter set giving the lowest average gap over 3 runs, a score of 8 is assigned to the parameter set giving the second lowest average gap, and so on. The total score assigned to a parameter set is the sum of scores assigned to this set for each problem. Average gaps and scores for each parameter set are summarized in Table 4.1.

The results displayed in Table 4.1 seem to indicate that increasing the dimension of the reference set is not necessarily beneficial. In fact, the results point to the parameter set $|\mathcal{R}| = 6$ and $PRstart = 20$ as the most effective : it offers the lowest average gap and the highest score. This setting was used in all the experiments reported in the rest of this section.

$\|\mathcal{R}\|$, $PRstart$	Av. Gap	score
6, 10	2.07%	60
8, 10	2.90%	39
15, 10	3.30%	28
6, 20	1.42%	75
8, 20	2.04%	53
15, 20	2.62%	42
6, 50	1.91%	55
8, 50	1.95%	64
15, 50	2.20%	49

TAB. 4.1 – Parameter Set Performances

4.4.2 Path Relinking Strategies

We tested all combinations of strategies in order to identify the best way to build the reference set (strategy S_i) and select initial and guiding solutions (strategy C_j). The best (S_i, C_j) combination was then used for the extensive experimental analysis of the path relinking procedure.

The same 10 problems used to calibrate the parameter settings were also used here. Moreover, each run is repeated 3 times. Thus, since there are 36 possible combinations of strategies, a total of 1080 runs were performed. All strategies have been run for the same number of iterations. Therefore, the performance of each combination of strategies is measured as the average improvement in solution quality, compared to the corresponding result of the cycle-based tabu search.

The comparative performance of all combinations of strategies is presented in Table 4.2. The first conclusion is that path relinking offers a consistently better performance in terms of solution quality than the local search heuristic. All combinations of strategies, except one, improve the results of cycle-based tabu search.

Table 4.2 identifies strategies *S3* and *C5* as offering the best results. This set was therefore retained for our experimental analyses.

The combination (*S3*, *C5*) offers a combination of very good solutions obtained over time - local optima better than the current best in $\mathcal{R}$ - and a possibly long exploration that proceeds towards the best solution in the reference set while starting from the solution most different from it. It is noteworthy that the second best couple

	C1	*C2*	*C3*	*C4*	*C(5)*	*C(6)*
S1	-0.51%	-0.91%	-0.84%	-0.47%	-0.55%	-0.42%
S2	-0.23%	-0.18%	-0.73%	-0.39%	-0.50%	-0.19%
S3	-0.54%	-0.51%	-0.93%	-0.69%	-0.99%	-0.80%
S4	-0.39%	-0.80%	-0.75%	-0.39%	-0.59%	-0.29%
S5	-0.36%	-0.69%	-0.61%	-0.41%	-0.45%	-0.43%
S6	-0.46%	-0.46%	-0.01%	-0.47%	0.08%	-0.16%

TAB. 4.2 – Average Improvement Over Cycle-based Tabu Search

of strategies, (*S3*, *C3*) is of a very similar structure. The success of these combinations confirms the importance of, on the one hand, the presence of elite solutions in the reference set and, on the other hand, of relinking strategies that diversify the solutions considered. This emphasizes the diversification effect of path relinking and supports some of the conclusions of Laguna and Armentano (2001).

It is interesting to examine the case of strategy *S1*. One would expect this strategy, which selects overall best solutions, to perform very well. This is not generally the case, however. One explanation may actually come from their very nature of overall best solutions generated by a local search procedure combined to the limited number of solutions included in $\mathcal{R}$. These solutions may share "too" many characteristics and not offer sufficient room for diversification. It is indeed noteworthy that the best performance of strategy *S1* is obtained jointly with strategy *C2* that builds a path from the second best to the best solution in $\mathcal{R}$. This couple, the third best overall, thus implements an enhanced intensification phase around the best solutions found by the local search.

The previous remarks underscore the capability of path relinking to implement both intensification and diversification phases. They also emphasize the need for further studies regarding the behaviour of strategies to build the reference set and to select the solutions to be linked. These studies are related to those regarding scatter search (Glover 1994, Glover and Laguna 1997, Glover, Laguna, and Marti 2000)

and other population-based meta-heuristics and are clearly beyond the scope of this paper.

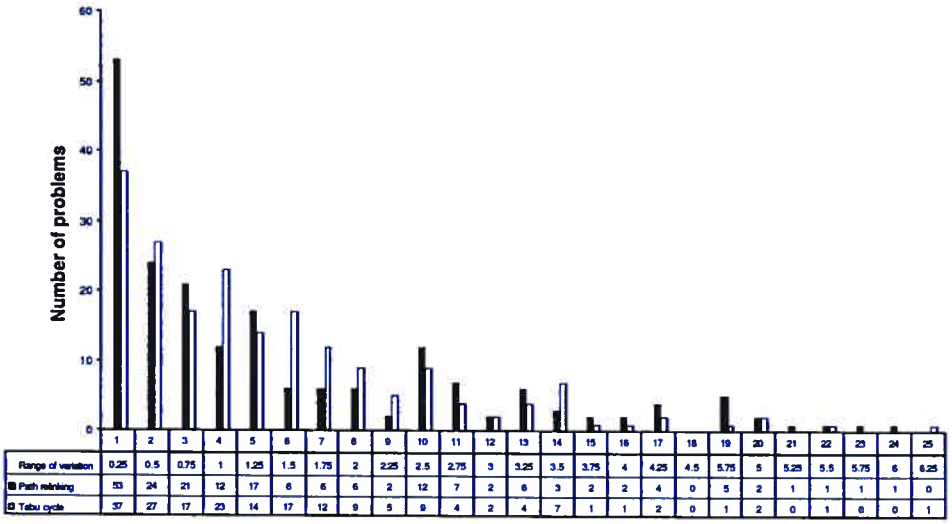
4.4.3 Influence of Sampling

We desire to investigate the impact of this element of randomness on the robustness of the cycle-based tabu search and path relinking procedures.

In general, one uses sampling in heuristic algorithms to save computation time. The corresponding price is that solutions produced in different runs are generally different. Relatively large differences usually signal a significant impact of the random factors and a somewhat less robust method. The solution of choice then is to perform multiple runs, to select the best solution. Of course, the corresponding running time of this strategy is the total computing time of all runs, which may rapidly become very onerous. On the other hand, when differences are small enough, the first solution that is produced may be considered as representative. The algorithm may then be considered robust and not sensitive to sampling.

We run both the path relinking and the cycle-based tabu search meta-heuristics 3 times on each of the 196 problems used for experimentation. For each problem, the relative gap between solutions produced over the 3 runs is computed as the percentage gap between the worst and the best solution relative to the worst solution.

Figure 4.9 displays the relative gap distributions for the two procedures. Relative gaps are grouped within intervals of 0.25 points of percentage. In each interval, the black and white columns are associated with path relinking and tabu search, respectively, and represent the number of problems with results in the specific range. Two conclusions emerge from the figures displayed in Figure 4.9. First, the relative gaps for both meta-heuristics are tight and the methods appear robust. In fact 74% of the problems solved with both meta-heuristics have a relative gap less than or equal to 2%. Second, path relinking is even less influenced by sampling and is more robust. This follows from the fact that sampling is replaced during the path relinking phase by a deterministic way to select the candidate links in $\mathcal{C}$.



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
Range of variation	0.25	0.5	0.75	1	1.25	1.5	1.75	2	2.25	2.5	2.75	3	3.25	3.5	3.75	4	4.25	4.5	5.75	5	5.25	5.5	5.75	6	6.25
Path relinking	53	24	21	12	17	6	6	6	2	12	7	2	6	3	2	2	4	0	5	2	1	1	1	1	0
Tabu cycle	37	27	17	23	14	17	12	9	5	9	4	2	4	7	1	1	2	0	1	2	0	1	0	0	1

FIG. 4.9 – Influence of Sampling

4.4.4 Result Analysis

To evaluate the performance of the path relinking algorithm proposed in this paper, we compare its output to the results of the cycle-based tabu search (Ghamlouche, Crainic, and Gendreau 2003) and to the optimal solutions obtained using the branch-and-bound algorithm of CPLEX 6.5 (1999).

The same two data sets of networks used by Ghamlouche, Crainic, and Gendreau (2003) were also used to test the path relinking method. Problems in both sets are general transshipment networks with no parallel arcs. Each commodity corresponds to a single origin-destination pair. On each arc, routing costs are the same for all commodities. Problem instances have been generated to offer for each network size a variety of fixed cost to routing cost ratios and capacity to demand ratios. detailed description of problem instances is given in Crainic, Frangioni, and Gendron (2001; see also Gendron and Crainic 1994, 1996). The problem generators as well as the problem instances can be obtained from the authors.

The first set of problem instances, denoted **C**, contains 43 problems defined by the number of nodes (20, 25, 30, 100), arcs (100, 230, 300, 400, 520, 700), and commodities (10, 30, 40, 100, 200, 400), as well as two letters summarizing the fixed cost and capacity information : a relatively high or low fixed cost relative to the routing cost is signaled by the letter **F** or **V**, respectively, while letters **T** and **L** indicate respectively whether the problem is tightly or somewhat loosely capacitated compared to the total demand. Detailed results for this set are presented in this section.

Tables 4.3 and 4.4 display results for set **C**. Problems are identified in the first column by the number of nodes, arcs, and commodities, as well as two letters summarizing the fixed cost and capacity information. The OPT column corresponds to the solution of the branch-and-bound algorithm solved using CPLEX 6.5 (1999) on the same computer. A limit of 10 hours was imposed. An **X** indicates that the procedure has failed to produce a feasible solution within this time limit, while a **t** indicates that the procedure stopped due to a time limit condition.

To facilitate comparisons in terms of solution quality and computation time, the cycle-based tabu search and the path relinking algorithm were run 3 times for each

PROB	OPT	TC	PR	AV. TC	AV. PR	IMPROV
25,100,10,V,L	14712	14712	14712	14769.33	14712	-0.39%
	(0.36)	(19.48)	(12.54)	(19.48)	(12.33)	
25,100,10,F,L	14941	14941	14941	14941	15081.33	0.94%
	(53.64)	(22.33)	(14.13)	(22.32)	(14.14)	
25,100,10,F,T	49899	50529	49899	50619.67	50154	-0.92%
	(40.58)	(33.84)	(24.06)	(33.12)	(23.52)	
25,100,30,V,T	365272	365385	365385	365385	365385	0.00%
	(16.62)	(141.51)	(101.44)	(142.37)	(103.40)	
25,100,30,F,L	37055	37515	37654	38672.33	37704	-2.50%
	(1727.96)	(112.48)	(75.15)	(112.48)	(73.91)	
25,100,30,F,T	85530	87325	86428	88095.33	86492.33	-1.82%
	(534.18)	(132.92)	(96.98)	(130.95)	(97.45)	
20,230,40,V,L	423848	430628	424385	430768.33	424789.67	-1.39%
	(10.43)	(214.12)	(148.82)	(217.16)	(145.90)	
20,230,40,V,T	371475	372522	371811	372888	372162.33	-0.19%
	(52.37)	(241.44)	(156.92)	(239.58)	(160.36)	
20,230,40,F,T	643036	652775	645548	657803.67	646378.67	-1.74%
	(671.76)	(259.53)	(172.16)	(259.53)	(163.47)	
20,300,40,V,L	429398	432007	429398	432168.33	429523.67	-0.61%
	(3.76)	(304.60)	(224.91)	(307.50)	(218.45)	
20,300,40,F,L	586077	602180	590427	604072.67	594590.67	-1.57%
	(145.55)	(335.59)	(228.33)	(341.06)	(234.53)	
20,300,40,V,T	464509	466115	464509	466546.33	464822.67	-0.37%
	(83.88)	(378.70)	(247.89)	(378.42)	(244.04)	
20,300,40,F,T	604198	615426	609990	616859.67	609995.33	-1.11%
	(59.88)	(349.48)	(214.43)	(354.30)	(219.52)	
20,230,200,V,L	94386	100001	100404	101481.33	101469.33	-0.01%
	(t)	(2585.94)	(2494.92)	(2596.49)	(2502.82)	
20,230,200,F,L	141737.40	148066	147988	148975	151352	1.60%
	(t)	(3141.91)	(2878.27)	(3141.91)	(2760.73)	
20,230,200,V,T	97914	106868	104689	107589.33	105598.67	-1.85%
	(t)	(2729.57)	(2210.86)	(2729.57)	(2304.53)	
20,230,200,F,T	137271	147212	147554	147868	148044.33	0.12%
	(t)	(3634.10)	(3385.75)	(3656.85)	(3505.63)	
20,300,200,V,L	74972.40	81367	78184	82186.67	79095.33	-3.76%
	(t)	(4085.89)	(3565.98)	(4007.80)	(3649.88)	
20,300,200,F,L	117306	122262	123484	123247.67	124924.67	1.36%
	(t)	(4210.14)	(4012.64)	(4336.93)	(4173.73)	
20,300,200,V,T	74991	80344	78866.80	80724	79212.27	-1.87%
	(t)	(4203.84)	(3924.21)	(4140.07)	(3714.29)	
20,300,200,F,T	108252	113947	113584	115342	114632.33	-0.62%
	(t)	(4854.61)	(3857.14)	(4643.75)	(3486.90)	

TAB. 4.3 – Computational Results, C problems

PROB	OPT	TC	PR	AV.TC	AV.PR	IMPROV
100,400,10,V,L	28423	28786	28485	28836.67	28529	-1.07%
	(84.81)	(252.15)	(89.21)	(254.99)	(83.97)	
100,400,10,F,L	24436	24022	24022	24022	24022	0.00%
	(t)	(196.50)	(82.86)	(201.38)	(79.20)	
100,400,10,F,T	66364	67184	65278	67376.33	65539.33	-2.73%
	(t)	(451.28)	(209.93)	(474.44)	(177.84)	
100,400,30,V,T	385544	385508	384926	385512.67	385181.67	-0.09%
	(t)	(1199.88)	(492.76)	(1220.42)	(496.29)	
100,400,30,F,L	50496	51831	51325	52176.33	51875.67	-0.58%
	(t)	(717.42)	(314.97)	(735.50)	(328.19)	
100,400,30,F,T	141278	147193	141359	147478	143403.67	-2.76%
	(t)	(1300.79)	(480.86)	(1300.79)	(522.44)	
30,520,100,V,L	53966	56603	54904	56722	55043.67	-2.96%
	(t)	(2260.69)	(1194.12)	(2260.69)	(1199.35)	
30,520,100,F,L	95294	103657	102054	105276.33	104222.33	-1.00%
	(t)	(2683.74)	(1459.99)	(2623.55)	(1508.39)	
30,520,100,V,T	52085	54454	53017	54537	53212	-2.43%
	(t)	(2715.76)	(1513.66)	(2711.55)	(1402.92)	
30,520,100,F,T	98357	105130	106130	107885.33	107575	-0.29%
	(t)	(2891.93)	(1522.68)	(2877.52)	(1478.54)	
30,700,100,V,L	47603	50041	48723	50098.33	48894.33	-2.40%
	(1736.05)	(2959.51)	(1860.61)	(2959.51)	(1815.84)	
30,700,100,F,L	60525	64581	63091	64802.67	63436.33	-2.11%
	(t)	(3181.72)	(1837.50)	(3181.72)	(1754.95)	
30,700,100,V,T	45944.50	48176	47209	48249.67	47288.33	-1.99%
	(t)	(3745.82)	(1894.08)	(3688.35)	(1832.10)	
30,700,100,F,T	55709	57628	56575.50	58111.33	56808.83	-2.24%
	(t)	(3547.09)	(1706.06)	(3547.09)	(1713.94)	
30,520,400,V,L	112997.50	122673	119416	123277.33	119624	-2.96%
	(t)	(55771.20)	(27477.40)	(55720.50)	(33716.57)	
30,520,400,F,L	X	164140	163112	165458	163377	-1.26%
	(t)	(40070.40)	(36669.30)	(41567.30)	(36600.43)	
30,520,400,V,T	X	122655	120170	123210.33	120764.33	-1.99%
	(t)	(4678.80)	(23089.10)	(21437.40)	(25705.77)	
30,520,400,F,T	X	169508	163675	170301.33	164921.33	-3.16%
	(t)	(49886.80)	(52173.20)	(51400.35)	(44862.23)	
30,700,400,V,L	X	107727	105116	108459.33	105403.33	-2.82%
	(t)	(38856.90)	(22314.80)	(36282.25)	(19742.33)	
30,700,400,F,L	X	150256	145026	150909	147887.33	-2.00%
	(t)	(68214.20)	(75664.90)	(73899.05)	(66489.40)	
30,700,400,V,T	X	101749	101212	103112.33	102119.33	-0.96%
	(t)	(51764.10)	(24288.90)	(49239.65)	(28664.70)	
30,700,400,F,T	X	144852	141013	146705	141446.67	-3.58%
	(t)	(79053.40)	(44936.40)	(79053.40)	(49824.83)	

TAB. 4.4 – Computational Results, C problems

problem. Columns TC and AV.TC hold respectively the best and the average solution of the cycle-based tabu search, while columns PR and AV.PR display the best and the average solution found by the path relinking approach, respectively. The figures in parentheses in the first three columns represent total computation times in CPU seconds. The corresponding figures in columns AV.TC and AV.PR represent the average total computation times in CPU seconds. Column IMPROV displays the percentage of improvement of the average path relinking solution relative to that of the cycle-based tabu search (i.e, AV.TC).

Two main observations emerge from the analysis of results presented in Tables 4.3 and 4.4. First, the path relinking procedure displays a satisfactory behavior. Comparing columns TC and PR, one observes that path relinking obtains the optimal solutions for 5 problems compared to 2 for TC and improves the best solutions found by cycle-based tabu search for 37 problems out of 41. The same behavior is also observed when average solution values are compared : the overall performance is clearly superior for the path relinking approach. The performance of path relinking also appears consistent over the range of problems, including relatively large ones in terms of number of arcs (e.g., 700) and commodities (e.g., 400). When branch-and-bound failed to identify a feasible solution within 10 hours, the proposed algorithm has identified best-known solutions for these problems with a reasonable computational effort.

The second observation concerns the computing effort. Indeed, despite of the fact that the path relinking procedure is stopped on the same criterion as the cycle-based tabu search, 400 iterations, Tables 4.3 and 4.4 indicate that the computational time for the former is often significantly shorter compared to that of the latter. This follows principally from the reduced number of resolutions of the multicommodity capacitated network flow problem and the fact that the dimension of the neighbourhood used during the path relinking phase is smaller.

The interesting performance and behavior of the proposed path relinking procedure is confirmed by the results obtained on the second set of problems instances, denoted **R**. There are 153 problems divided in 18 groups. Each group contains the same number of nodes, arcs, and commodities but with different combined level of fixed cost and capacity ratios. Three levels of fixed cost and capacity ratios are consi-

dered : F01 = 0.01, F05 = 0.05, and F10 = 0.10 indicating continuously higher levels of fixed costs compared to routing costs, and C1 = 1, C2 = 2, and C8 = 8 that signal that the total capacity available becomes increasingly tighter relatively to the total demand (the formulas used to compute these ratios are presented in Ghamlouche, Crainic, and Gendreau (2003). Thus, the easiest problems generally have F01 and C1 ratios, while the most difficult ones have F10 and C8 characteristics. Only aggregated statistics are used in the following to support the discussion. Detailed results are included in the Appendix.

Table 4.5 displays the distribution of the optimality gaps relative to CPLEX branch-and-bound for both the cycle-based tabu search and the path relinking procedures. Values are calculated for both the best and average solutions for the C and R problems. Column PSet identifies the problems set, the X column indicates the number of problems where branch-and-bound did not find a feasible solution (in 10 hours), the OPT column indicates the number of optimal solutions found by the meta-heuristics, and the IMP column displays the number of problems where the meta-heuristic solution is better than the one found by branch-and-bound after 10 hours of computation. The next six columns correspond to the indicated gap intervals.

	PSet	X	OPT	IMP	(0-2%]	(2-4%]	(4-6%]	(6-8%]	(8-10%]	>10%
TC	C	7	2	2	11	4	9	4	4	-
PR	C	7	5	3	14	2	7	5	-	-
TC	R	-	23	-	48	31	20	14	11	6
PR	R	-	38	-	56	26	14	11	4	4
AV.TC	C	7	1	2	9	5	9	5	4	1
AV.PR	C	7	1	3	16	4	5	5	2	-
AV.TC	R	-	19	-	43	27	25	17	10	12
AV.PR	R	-	20	-	61	29	17	10	8	8

TAB. 4.5 – Distribution of Relative Gaps

The results displayed in Table 4.5 show that path relinking obtains a larger number of optimal solutions than the cycle-based tabu search. Moreover, the distribution of the gaps indicates that path relinking solutions are closer to the optimum than those of the cycle-based tabu search, in terms of both best and average solution values. On average, for problems in sets **C** and **R**, path relinking obtains gaps of 2.32%

and 3.08%, respectively, of the best solutions found by branch-and-bound. But then, branch-and-bound cannot address all problems. Compared to the cycle-based tabu search, the average improvement obtained by path relinking is 1.35% and 0.83% for problems in sets **C** and **R**, respectively, while the maximum improvement is 3.76% and 4.75%, respectively. In fact, when a different solution was found, the path relinking improves the best solutions found by the cycle-based tabu search for 117 out of 140 problems in set **R**.

Table 4.6 displays the optimality gap distribution of the best and average solutions of the cycle-based tabu search and path relinking according to fixed cost and capacity ratios for problems in set **R**. Table 4.7 displays the same information but according to problem dimensions. The results displayed in these tables underscore the important role of the path relinking phase in the search. In all cases, the path relinking procedure achieves better results than the cycle-based tabu search and reduces significantly the optimality gap. This behavior is consistent over the entire range of problem dimensions, number of commodities, and variation of fixed cost and capacity. Consequently, we believe that the approach proposed performs well and is able to handle variation in problem dimensions and structure.

	C1		C2		C8	
	TC	PR	TC	PR	TC	PR
F01	1.33%	0.76%	1.39%	0.78%	1.91%	1.15%
F05	3.32%	2.43%	3.65%	2.64%	4.43%	3.23%
F10	4.15%	3.09%	4.35%	3.04%	5.25%	4.11%
	C1		C2		C8	
	AV.TC	AV.PR	AV.TC	AV.PR	AV.TC	AV.PR
F01	1.69%	0.99%	1.70%	1.00%	2.17%	1.44%
F05	4.20%	3.27%	4.20%	3.49%	5.20%	4.15%
F10	5.00%	4.05%	5.36%	4.14%	6.32%	5.50%

TAB. 4.6 – Gap Distribution According to Fixed Cost and Capacity Level, **R** Problems

4.5 Conclusion

We have presented a path relinking algorithm for the fixed-charge, capacitated, multicommodity network design problem. Elite solutions are constructed by performing a simple tabu search algorithm using cycle-based neighbourhood structures. A

$\|P\|,\|N\|,\|A\|$	TC	PR	$\|P\|,\|N\|,\|A\|$	TC	PR	$\|P\|,\|N\|,\|A\|$	TC	PR
10,10,25	0.00%	0.00%	10,10,50	0.23%	0.08%	10,10,75	0.70%	0.04%
25,10,25	1.41%	0.23%	25,10,50	0.82%	0.36%	25,10,75	1.92%	0.41%
50,10,25	1.48%	0.61%	50,10,50	2.36%	1.14%	50,10,75	2.69%	1.52%
40,20,100	2.50%	1.37%	40,20,200	4.37%	3.59%	40,20,300	3.91%	2.08%
100,20,100	2.98%	2.05%	100,20,200	5.36%	4.93%	100,20,300	6.99%	4.68%
200,20,100	4.80%	4.55%	200,20,200	6.76%	5.41%	200,20,300	7.39%	6.84%
$\|P\|,\|N\|,\|A\|$	AV.TC	AV.PR	$\|P\|,\|N\|,\|A\|$	AV.TC	AV.PR	$\|P\|,\|N\|,\|A\|$	AV.TC	AV.PR
10,10,25	0.05%	0.00%	10,10,50	0.36%	0.12%	10,10,75	0.89%	0.46%
25,10,25	1.70%	0.60%	25,10,50	1.16%	0.55%	25,10,75	2.47%	0.92%
50,10,25	1.92%	0.72%	50,10,50	3.06%	1.93%	50,10,75	3.37%	2.30%
40,20,100	3.34%	1.70%	40,20,200	5.52%	4.84%	40,20,300	4.80%	3.91%
100,20,100	3.52%	2.82%	100,20,200	6.47%	5.90%	100,20,300	7.82%	5.79%
200,20,100	5.29%	4.86%	200,20,200	7.30%	6.42%	200,20,300	9.32%	8.97%

TAB. 4.7 – Gap Distribution According to Problem Dimensions, **R** Problems

restricted version of the same cycle-based neighbourhoods is used to build the path linking two elite solutions in the reference set. We have proposed and compared several strategies to build the reference set and to select initial and guiding solutions. The performance of the algorithm was evaluated using 196 problem instances of several types and sizes.

The results are very satisfactory. The basic idea of combining the cycle-based tabu search and path relinking is a successful one. In most cases, the path relinking achieves better results. Moreover, the algorithm appears robust in terms of solution quality and computational effort. To our knowledge, in fact, the proposed path relinking meta-heuristic offers the best current performance among approximate solution methods for the fixed-charge, capacitated, multicommodity network design problem.

With respect to the path relinking methodology, this paper enriches the rather limited trove of research results reported in the literature. It reinforces, in particular, the current guidelines relative to how to build the reference set and to select the solutions to be linked. The paper also underscores the capability of path relinking to implement both search intensification and diversification phases.

The experimental results have also pointed out a number of interesting research perspectives. One such avenue is concerned with the criteria used to select and combine solutions in population-based meta-heuristics, such as path relinking, scatter

search, and more traditional genetic algorithms. This issue is especially important when solutions display a mixed integer nature with complex network structures and sub-problems, as is the case for the problems we address. A second research direction concerns the mechanisms to learn (memories) and guide the various phases of the search - local search, diversification, intensification, and so on -, the relations to the already-mentioned population-based methods, and the combinations of these various elements into a comprehensive meta-heuristic for the fixed-charge, capacitated, multicommodity network design problem. We intend to report development on both subjects in the relatively close future.

References

Balakrishnan, A., Magnanti, T.L., and Mirchandani, P. (1997). Network Design. In Dell'Amico, M., Maffioli, F., and Martello, S., editors, *Annotated Bibliographies in Combinatorial Optimization*, pages 311–334. John Wiley & Sons, New York, NY.

Crainic, T.G., Frangioni, A., and Gendron, B. (2001). Bundle-Based Relaxation Methods for Multicommodity Capacitated Network Design. *Discrete Applied Mathematics*, 112 :73–99.

Crainic, T.G., Gendreau, M., and Farvolden, J.M. (2000). A Simplex-Based Tabu Search Method for Capacitated Network Design. *INFORMS Journal on Computing*, 12(3) :223–236.

Gendron, B., Crainic, T.G., and Frangioni, A. (1998). Multicommodity Capacitated Network Design. In Sansó, B. and Soriano, P., editors, *Telecommunications Network Planning*, pages 1–19. Kluwer Academic Publishers, Norwell, MA.

Gendron, B. and Crainic, T.G. (1994). Relaxations for Multicommodity Network Design Problems. Publication CRT-965, Centre de recherche sur les transports, Université de Montréal, Montréal, QC, Canada.

Gendron, B. and Crainic, T.G. (1996). Bounding Procedures for Multicommodity Capacitated Network Design Problems. Publication CRT-96-06, Centre de recherche sur les transports, Université de Montréal, Montréal, QC, Canada.

Ghamlouche, I., Crainic, T.G., and Gendreau, M. (2003). Cycle-based Neighbourhoods for Fixed-Charge Capacitated Multicommodity Network Design. *Operations Research*, 51(4) :655–667.

Glover, F., Laguna, M., and Marti, R. (2000). Fundamentals of scatter search and path relinking. *Control and Cybernetics*, 39(3) :653–684.

Glover, F. and Laguna, M. (1997). *Tabu Search*. Kluwer Academic Publishers, Norwell, MA.

Glover, F. (1994). Genetic Algorithms and Scatter Search : Unsuspected Potentials. *Statistics and Computing*, 4 :131–140.

Glover, F. (1997). A Template for Scatter Search and Path Relinking. In Hao, J., Lutton, E., Ronald, E., Schoenauer, M., and Snyers, D., editors, *Artificial Evolution*, volume 1363 of *Lecture Notes in Computer Science*, pages 13–54. Springer Verlag, Berlin.

Holmberg, K. and Yuan, D. (2000). A Lagrangean Heuristic Based Branch-and-Bound Approach for the Capacitated Network Design Problem. *Operations Research*, 48(3) :461–481.

ILOG (1999). *ILOG CPLEX 6.5*. ILOG, Mountain View, CA. U.S.A.

Laguna, M. and Armentano, V.A. (2001). Lessons from Applying and Experimenting with Scatter Search. In Rego, C. and Alidaee, B., editors, *Adaptive Memory and Evolution : Tabu Search and Scatter Search*. Kluwer Academic Publishers, Norwell, MA. to appear.

Laguna, M., Marti, R., and Campos, V. (1999). Intensification and Diversification with Elite Tabu Search Solutions for the Linear Ordering Problem. *Computers & Operations Research*, 22 :1217–1230.

Laguna, M. and Marti, R. (1999). GRASP and Path Relinking for 2-Layer Straight Line Crossing Minimization. *INFORMS Journal on Computing*, 11(1) :44–52.

Magnanti, T.L. and Wong, R.T. (1984). Network Design and Transportation Planning : Models and Algorithms. *Transportation Science*, 18(1) :1–55.

Minoux, M. (1989). Network Synthesis and Optimum Network Design Problems : Models, Solution Methods and Applications. *Networks*, 19 :313–360.

Ribeiro, C.C., Uchoa, E., and Werneck, R (2002). A Hybrid GRASP with Pertur-

bations for the Steiner Problem in Graphs. *INFORMS Journal on Computing*, 14(3) :228–246.

Appendix

Problems **R** were generated to facilitate the study of the impact of particular problem characteristics on algorithmic performance. The number of nodes, design arcs, and commodities are systematically varied, one at a time, and for each of these networks, nine problem instances were created by combining three levels of fixed cost ratios and three levels of capacity ratios.

The *fixed cost ratio* is computed as $|\mathcal{P}| \sum_{(ij)\in\mathcal{A}} f_{ij} / \sum_{p\in\mathcal{P}} w^p \sum_{(ij)\in\mathcal{A}} c_{ij}^p$, and the three values considered are F01 = 0.01, F05 = 0.05, and F10 = 0.10 corresponding to continuously higher levels of fixed costs compared to the routing costs. The *capacity ratio* is is computed as $|\mathcal{A}| \sum_{p\in\mathcal{P}} w^p / \sum_{(ij)\in\mathcal{A}} u_{ij}$, and the values considered are C1 = 1, C2 = 2, and C8 = 8, indicating that the total capacity available becomes increasingly tighter relatively to the total demand. Thus, the easiest problems generally have F01 and C1 ratios, while the most difficult ones have F10 and C8 characteristics.

Results for each problem instance of set **R** are given in Tables 4.8 to 4.25. The following information is displayed :

- OPT : The optimal solution obtained by using the branch-and-bound algorithm of CPLEX version 6.5. A (t) indicates that the procedure stopped due to a time limit condition. A limit of 10 hours CPU time was imposed.
- TC and PR : The best solution obtained respectively by the cycle-based tabu search and path relinking over 3 runs.
- AV.TC and AV.PR : The average solution obtained respectively by the cycle-based tabu search and path relinking over 3 runs.
- IMPROV : The average improvement in solution quality between the average solution of path relinking and that of the cycle-based tabu search.

All tests were conducted on one 400MHz processor of a 64-processors Sun Enterprise 10000 with 64 Gigabyte of RAM, operating under Solaris 2.7.

The figures in parentheses in the first 3 columns represent total computation times in CPU seconds, while the corresponding figures in columns AV.TC and AV.PR represent average total computation times.

TAB. 4.8 – Computational Results R01 : 10,25,10

PROB	OPT	TC	PR	AV.TC	AV.PR	IMPROV
R01,F01,C1	74079	74079	74079	74079	74079	0%
	(0.04)	(6.53)	(6.66)	(6.66)	(6.43)	
R01,F05,C1	92403	92403	92403	92403	92403	0%
	(0.04)	(6.13)	(4.95)	(6.37)	(6.13)	
R01,F10,C1	115304	115304	115304	115304	115304	0%
	(0.09)	(6.28)	(4.76)	(6.40)	(5.32)	
R01,F01,C2	84908	84908	84908	85066.67	84908	-0.19%
	(0.46)	(7.62)	(6.39)	(8.70)	(6.34)	
R01,F05,C2	113036	113036	113036	113036	113036	0%
	(1.35)	(9.84)	(7.42)	(9.36)	(7.45)	
R01,F10,C2	147599	147599	147599	147796.67	147599	-0.13%
	(2.36)	(8.62)	(7.74)	(9.16)	(7.84)	

TAB. 4.9 – Computational Results R02 : 10,25,25

PROB	OPT	TC	PR	AV.TC	AV.PR	IMPROV
R02,F01,C1	232239	232239	232239	232239	232239	0%
	(0.31)	(16.21)	(13.03)	(17.48)	(14.76)	
R02,F05,C1	322453	328005	326597	330429.00	327066.33	-1.02%
	(1.12)	(24.67)	(20.09)	(24.89)	(23.56)	
R02,F10,C1	419503	439687	419503	440508.00	426999.66	-3.07%
	(1.37)	(25.31)	(20.69)	(25.51)	(26.43)	
R02,F01,C2	316437	316437	316437	316437.00	316437	0%
	(0.23)	(25.55)	(21.92)	(25.86)	(22.12)	
R02,F05,C2	431250	432234	431734	433979.00	432470	-0.35%
	(0.53)	(29.45)	(23.28)	(29.85)	(26.34)	
R02,F10,C2	559578	569032	559578	571400.67	560082.33	-1.98%
	(0.55)	(27.32)	(23.56)	(29.06)	(24.56)	

TAB. 4.10 – Computational Results R03 :10,25,50

PROB	OPT	TC	PR	AV.TC	AV.PR	IMPROV
R03,F01,C1	484830	484830	484830	484830	485678.33	0.17%
	(0.67)	(52.46)	(58.16)	(53.26)	(54.23)	
R03,F05,C1	703362	716524	704893	718578.33	706213	-1.72%
	(2.01)	(60.65)	(58.17)	(66.84)	(60.23)	
R03,F10,C1	944990	985909	962408	992413.67	964399	-2.82%
	(2.34)	(75.81)	(58.25)	(76.22)	(60.12)	
R03,F01,C2	704247	705207	705207	705207.00	705207	0%
	(0.47)	(89.35)	(46.86)	(89.88)	(50.23)	
R03,F05,C2	932897	942929	939055	943995.00	939372.66	-0.49%
	(1.26)	(89.41)	(67.14)	(90.34)	(70.23)	
R03,F10,C2	1188638	1205740	1197950	1224473.33	1198550	-2.12%
	(0.98)	(90.14)	(60.16)	(91.32)	(64.12)	

TAB. 4.11 – Computational Results R04 :10,50,10

PROB	OPT	TC	PR	AV.TC	AV.PR	IMPROV
R04,F01,C1	31730	31730	31730	31730	31730	0%
	(0.02)	(9.51)	(6.84)	(9.59)	(7.4)	
R04,F05,C1	48920	48920	48920	48920	48920	0%
	(0.06)	(8.85)	(6.28)	(9.52)	(7.34)	
R04,F10,C1	63767	63767	63767	63767	63767	0%
	(0.06)	(9.75)	(7.36)	(10.15)	(8.43)	
R04,F01,C2	33740	33740	33740	33740	33740	0%
	(0.57)	(9.85)	(7.4)	(10.19)	(876)	
R04,F05,C2	53790	53790	53790	53790	53790	0%
	(2.62)	(10.39)	(7.62)	(10.65)	(7.45)	
R04,F10,C2	74030	74030	74030	74030	74030	0%
	(1.72)	(10.37)	(8.18)	(10.76)	(8.46)	
R04,F01,C8	68291.7	68517	68310	68551.67	68456	-0.14%
	(5.14)	(13.27)	(11.45)	(14.53)	(12.53)	
R04,F05,C8	113004	113226	113226	113744	113226	-0.46%
	(9.41)	(15.79)	(11.69)	(15.99)	(12.56)	
R04,F10,C8	163208	165657	164091	166764.67	164317	-1.47%
	(7.61)	(15.95)	(11.54)	(16.27)	(13.34)	

TAB. 4.12 – Computational Results R05 :10,50,25

PROB	OPT	TC	PR	AV.TC	AV.PR	IMPROV
R05,F01,C1	123003	123109	123003	123465.67	123199.66	-0.22%
	(0.38)	(23.48)	(17.94)	(23.61)	(20.21)	
R05,F05,C1	170060	172517	170467	173327.67	171573	-1.01%
	(1.51)	(24.24)	(18.4)	(24.44)	(18.45)	
R05,F10,C1	221486	225950	221486	226440.00	221967.33	-1.98%
	(8.44)	(25.57)	(19.78)	(25.70)	(20.4)	
R05,F01,C2	131608	131888	131608	132153.33	131662.66	-0.37%
	(1.58)	(26.55)	(20.38)	(26.89)	(20.67)	
R05,F05,C2	204157	206497	205764	207021.33	205764	-0.61%
	(15.60)	(28.49)	(21.53)	(28.90)	(23.43)	
R05,F10,C2	286524	290574	290574	293937.67	292887.66	-0.36%
	(65.12)	(28.57)	(22.9)	(30.12)	(22.8)	
R05,F01,C8	278372	278372	278372	278372	278372	0%
	(1.45)	(43.70)	(47.25)	(52.08)	(49.32)	
R05,F05,C8	445810	448542	449133	449319.00	446986.33	-0.52%
	(1.99)	(50.45)	(45.83)	(53.83)	(47.9)	
R05,F10,C8	625879	628628	626291	630216.00	628123.66	-0.33%
	(2.20)	(42.50)	(39.65)	(46.58)	(40.45)	

TAB. 4.13 – Computational Results R06 :10,50,50

PROB	OPT	TC	PR	AV.TC	AV.PR	IMPROV
R06,F01,C1	245936	247829	246365	248527.67	247028.66	-0.60%
	(3.97)	(58.64)	(48.84)	(59.55)	(49.87)	
R06,F05,C1	401685	410457	405672	414565.67	407483.66	-1.71%
	(51.12)	(71.69)	(57.08)	(72.96)	(60.12)	
R06,F10,C1	559477	587827	572112	592116.33	576620.33	-2.62%
	(205.18)	(69.56)	(61.74)	(73.04)	(59.24)	
R06,F01,C2	286682	288200	287510	288861.67	287551	-0.45%
	(25.18)	(74.92)	(62.51)	(79.97)	(62.4)	
R06,F05,C2	498266	517614	501560	519212.67	512575.33	-1.28%
	(463.4)	(94.42)	(71.95)	(97.03)	(73.67)	
R06,F10,C2	734414	775792	753337	783014.67	763504	-2.49%
	(1855.94)	(99.10)	(74.51)	(100.03)	(71.08)	
R06,F01,C8	682921	683481	683614	683525.33	683692.66	0.02%
	(1.47)	(172.53	(125.96)	(177.05)	(126.87)	
R06,F05,C8	1030479	1042510	1038600	1044586.67	1041900	-0.26%
	(1.45)	(176.18)	(130.92)	(190.94)	(133.98)	
R06,F10,C8	423316	431329	433690	442105.33	440456.33	-0.37%
	(46.56)	(54.54)	(45.34)	(56.35)	(45.45)	

TAB. 4.14 – Computational Results R07 :10,75,10

PROB	OPT	TC	PR	AV.TC	AV.PR	IMPROV
R07,F01,C1	32807	32807	32807	32807	32807	0%
	(0.08)	(15.65)	(14.00)	(14.99)	(13.6)	
R07,F05,C1	47252	47252	47252	47252	47252	0%
	(0.13)	(12.37)	(8.8)	(12.61)	(9.21)	
R07,F10,C1	62962	62962.0	62962	62962	62987.66	0.04%
	(0.62)	(15.17)	(9.62)	(15.42)	(10.34)	
R07,F01,C2	37432	37432.0	37432	37432	37432	0%
	(0.66)	(13.76)	(10.06)	(13.89)	(10.45)	
R07,F05,C2	56475	56841.0	56475	56841	56552.33	-0.51%
	(4.39)	(15.05)	(10.74)	(15.18)	(11.43)	
R07,F10,C2	77249	77863.0	77249	78842.67	78489.66	-0.45%
	(19.01)	(14.20)	(10.61)	(15.32)	(10.45)	
R07,F01,C8	59947	59947.0	59947	59947	60061.66	0.19%
	(11.05)	(17.99)	(15.03)	(19.62)	(18.62)	
R07,F05,C8	99194	102203.	99502	102747.33	99757.33	-2.91%
	(21.75)	(20.63)	(14.52)	(21.26)	(14.23)	
R07,F10,C8	141692	144329.	141788	144079	143898.33	-0.13%
	(54.39)	(19.16)	(14.84)	(20.62)	(15.23)	

TAB. 4.15 – Computational Results R08 : 10,75,25

PROB	OPT	TC	PR	AV.TC	AV.PR	IMPROV
R08,F01,C1	102531	102556	102531	102880.00	102547.66	-0.32%
	(1.33)	(31.60)	(26.15)	(32.02)	(25.14)	
R08,F05,C1	143894	145434	143894	145818.67	143894	-1.32%
	(0.54)	(31.53)	(24.62)	(31.93)	(24.45)	
R08,F10,C1	182793	182793	182793	182969.00	184068.33	0.60%
	(1.35)	(31.83)	(26.29)	(31.91)	(25.34)	
R08,F01,C2	109325	110193	109325	110308.33	109358.33	-0.86%
	(5.45)	(34.28)	(25.45)	(34.64)	(26.45)	
R08,F05,C2	157047	160848	157901	161447.67	158039	-2.11%
	(34.23)	(35.76)	(27.7)	(36.36)	(28.8)	
R08,F10,C2	207540	214793	208325	214993.00	209766	-2.43%
	(34.23)	(34.52)	(27.95)	(35.79)	(27.7)	
R08,F01,C8	154160	156738	154327	157065.33	155292.33	-1.13%
	(75.38)	(54.46)	(38.12)	(55.01)	(39.13)	
R08,F05,C8	274866.5	282525	278443	287224.33	280858.33	-2.22%
	(404.49)	(53.88)	(42.52)	(55.15)	(41.15)	
R08,F10,C8	415793	436669	421292	444136.00	427881	-3.66%
	(776.56)	(54.40)	(41.33)	(56.67)	(43.8)	

TAB. 4.16 – Computational Results R09 : 10,75,50

PROB	OPT	TC	PR	AV.TC	AV.PR	IMPROV
R09,F01,C1	171512	171902	171572	172382.67	171655.33	-0.42%
	(2.33)	(71.94)	(55.89)	(74.39)	(56.45)	
R09,F05,C1	296712	305497	302448	308971.67	303722	-1.70%
	(32.65)	(66.57)	(66.94)	(76.45)	(66.56)	
R09,F10,C1	424266	435762	431584	440892.33	436532.66	-0.99%
	(218.77)	(73.17)	(64.81)	(80.40)	(65.56)	
R09,F01,C2	192736	193734	192736	194155.67	193203.66	-0.49%
	(10.2)	(80.08)	(67.11)	(83.27)	(69.13)	
R09,F05,C2	357318	368592	366418	373219.67	371407.66	-0.49%
	(143.32)	(90.21)	(76.96)	(99.48)	(79.45)	
R09,F10,C2	522187	555768	533927	560263.67	543955	-2.91%
	(1227.52)	(86.08)	(80.26)	(93.20)	(83.98)	
R09,F01,C8	345057	348732	346773	349684.33	348735.33	-0.27%
	(27.83)	(151.35)	(124.73)	(153.15)	(125.87)	
R09,F05,C8	646579	668296	657541	669379.00	665286.33	-0.61%
	(89.5)	(156.57)	(117.78)	(161.14)	(127.54)	
R09,F10,C8	951136	987258.	979672	992832.00	980308.66	-1.26%
	(310.59)	(155.66)	(123.66)	(159.08)	(133.98)	

TAB. 4.17 – Computational Results R10 : 20,100,40

PROB	OPT	TC	PR	AV.TC	AV.PR	IMPROV
R10,F01,C1	200087	200484	200147	200542.33	200340.33	-0.10%
	(10.72)	(104.95)	(75.22)	(106.74)	(76.34)	
R10,F05,C1	346813.5	350407	347459	352366.00	347835.66	-1.29%
	(788.2)	(110.20)	(85.65)	(116.93)	(88.56)	
R10,F10,C1	488015	502724	502724	505694.00	504601	-0.22%
	(563.29)	(127.87)	(84.8)	(128.99)	(84.56)	
R10,F01,C2	229196	231302	229196	232220.67	229363	-1.23%
	(184.31)	(128.79)	(90.6)	(134.77)	(97.56)	
R10,F05,C2	411664	430254	425230	433229.00	426363.66	-1.58%
	(5123.25)	(149.47)	(102.63)	(149.82)	(110.34)	
R10,F10,C2	609104	646107	628353	664652.33	633081.33	-4.75%
	(27260.53)	(158.59)	(109.82)	(159.23)	(109.54)	
R10,F01,C8	486895	490034	487235	491176.67	487923.33	-0.66%
	(391.33)	(390.26)	(330.31)	(398.56)	(343.89)	
R10,F05,C8	951056	977346	963914	980466.67	967753.33	-1.30%
	(555.52)	(286.08)	(274.29)	(315.74)	(284.76)	
R10,F10,C8	1421740	1469020	1439320	1491893.33	1448780	-2.89%
	(608.38)	(228.92)	(248.12)	(243.37)	(245.78)	

TAB. 4.18 – Computational Results R11 : 20,100,100

PROB	OPT	TC	PR	AV.TC	AV.PR	IMPROV
R11,F01,C1	714431	726585	720236	730686.33	724441	-0.85%
	(392.46)	(441.29)	(355.76)	(447.77)	(358.43)	
R11,F05,C1	1265985	1323620	1296050	1327736.67	1304816.66	-1.73%
	(t)	(489.85)	(510.67)	(601.70)	(528.12)	
R11,F10,C1	1846295	1933560	1940180	1969320	1951226.66	-0.92%
	(t)	(706.75)	(576.94)	(723.43)	(587.76)	
R11,F01,C2	870451	879445	875908	883224.67	877552	-0.64%
	(2960.13)	(819.49)	(550.56)	(830.33)	(555.43)	
R11,F05,C2	1623640	1715040	1674340	1721046.67	1687013.33	-1.98%
	(21739.5)	(1137.94)	(889.13)	(1153.43)	(890.33)	
R11,F10,C2	2414060	2593590	2546730	2600196.67	2606896.66	0.26%
	(t)	(1274.95)	(990.76)	(1332.79)	(1010.45)	
R11,F01,C8	2294912	2294912	2294912	2295296.67	2294912	-0.02%
	(31.51)	(880.22)	(1181.57)	(1088.83)	(1089.56)	
R11,F05,C8	3507100	3540730	3529530	3550923.33	3539613.33	-0.32%
	(79.79)	(1174.59)	(1079.67)	(1235.99)	(1102.99)	
R11,F10,C8	4579353	4616190	4592820	4645180	4653570	0.18%
	(48.51)	(827.84)	(777.86)	(849.09)	(812.45)	

TAB. 4.19 – Computational Results R12 : 20,100,200

PROB	OPT	TC	PR	AV.TC	AV.PR	IMPROV
R12,F01,C1	1639443	1700170	1668370	1706230	1677986.66	-1.66%
	(4436.74)	(2199.46)	(2434.57)	(2720.30)	(2453.3)	
R12,F05,C1	3403074.2	3741660	3717800	3767176.67	3728346.66	-1.03%
	(t)	(3188.77)	(2975.35)	(3477.48)	(3177.3)	
R12,F10,C1	5276171	6055670	6037960	6100926.67	6073666.66	-0.45%
	(t)	(4096.82)	(3245.49)	(4331.77)	(3476.4)	
R12,F01,C2	2303557	2331290	2326930	2332063.33	2325936.66	-0.26%
	(1739.79)	(6504.21)	(5539.91)	(7467.51)	(6432.43)	
R12,F05,C2	4669799	4942920	4954250	4982423.33	4978036.66	-0.09%
	(3145.59)	(7364.80)	(5709.11)	(7574.81)	(5634.98)	
R12,F10,C2	7100019	7562660	7638050	7653836.67	7660990	0.09%
	(1683.84)	(6060.43)	(4735.91)	(7704.07)	(5432.98)	
R12,F01,C8	7635270	7638050	7638050	7638050	7638050	0.00%
	(78.13)	(5336.99)	(4735.91)	(5787.92)	(4654.33)	
R12,F05,C8	10067742	10106800	10085000	10116466.67	10104800	-0.12%
	(61.55)	(3176.64)	(3315.88)	(3347.35)	(3412.5)	
R12,F10,C8	11967768	12064300	12042800	12083133.33	12063866.67	-0.16%
	(50.29)	(2459.84)	(2276.53)	(2503.33)	(2341.4)	

TAB. 4.20 – Computational Results R13 : 20,200,40

PROB	OPT	TC	PR	AV.TC	AV.PR	IMPROV
R13,F01,C1	142947	143778	142947	144437.33	143139.66	-0.90%
	(6.41)	(201.55)	(130.35)	(202.37)	(138.4)	
R13,F05,C1	263800	272019	273853	275961.33	273912	-0.74%
	(547.98)	(203.12)	(141.32)	(205.89)	(145.32)	
R13,F10,C1	365836	375628	378922	379687.67	387201.33	1.98%
	(116.62)	(214.58)	(145.79)	(221.23)	(147.34)	
R13,F01,C2	150977	151875	150977	153080	152617.66	-0.30%
	(90.14)	(208.55)	(143.09)	(209.60)	(141.23)	
R13,F05,C2	282682	290150	289706	293241.67	294079.33	0.29%
	(3551.86)	(225.07)	(142.83)	(231.63)	(144.76)	
R13,F10,C2	406790	430246	435724	435988.33	439691.33	0.85%
	(29695.59)	(246.71)	(159.9)	(252.66)	(164.34)	
R13,F01,C8	208088	213650	210898	214628.67	212134	-1.16%
	(t)	(319.67)	(183.14)	(326.24)	(197.23)	
R13,F05,C8	446997	485018	469035	491599.67	481539.66	-2.05%
	(t)	(379.88)	(182.03)	(383.64)	(189.45)	
R13,F10,C8	698280	787477	761505	801558.67	773893	-3.45%
	(t)	(391.99)	(232.08)	(403.09)	(245.76)	

TAB. 4.21 – Computational Results R14 : 20,200,100

PROB	OPT	TC	PR	AV.TC	AV.PR	IMPROV
R14,F01,C1	403414	416932	410430	418294.67	411263.66	-1.68%
	(495.72)	(561.79)	(413.86)	(567.40)	(412.56)	
R14,F05,C1	750091	775023	778492	792663.67	787719.66	-0.62%
	(t)	(598.24)	(503.86)	(644.45)	(576.32)	
R14,F10,C1	1078595	1133420	1127980	1160773.33	1133240	-2.37%
	(t)	(723.41)	(555.53)	(737.84)	(604.12)	
R14,F01,C2	437607	448725	444170	451845.33	445088.33	-1.50%
	(1327.37)	(633.53)	(463.33)	(637.85)	(475.23)	
R14,F05,C2	852330	890914	883241	897678.00	894028	-0.41%
	(t)	(731.71)	(580.59)	(777.93)	(590.43)	
R14,F10,C2	1230749.7	1309970	1335170	1332676.67	1340806.66	0.61%
	(t)	(818.25)	(608.59)	(833.51)	(673.12)	
R14,F01,C8	670064	698381	689762	701782.33	692596.33	-1.31%
	(t)	(1794.67)	(1294.26)	(2000.69)	(1576.34)	
R14,F05,C8	1643376	1782720	1740600	1791180.00	1771183.33	-1.12%
	(t)	(2516.48)	(1810.25)	(2717.66)	(1965.45)	
R14,F10,C8	2630332	2901140	2939990	2911496.67	3008416.66	3.33%
	(t)	(3223.01	(2555.49)	(3335.62)	(2875.45)	

TAB. 4.22 – Computational Results R15 : 20,200,200

PROB	OPT	TC	PR	AV.TC	AV.PR	IMPROV
R15,F01,C1	1000787	1044180	1027930	1053690	1034716.66	-1.80%
	(5384.72)	(1698.44)	(1361.73)	(1808.50)	(1587.43)	
R15,F05,C1	1979413	2164870	2135640	2183736.67	2183433.33	-0.01%
	(t)	(2323.56)	(1871.08)	(2475.63)	(1896.43)	
R15,F10,C1	2949264	3156520	3131640	3178423.33	3190583.33	0.38%
	(t)	(2546.50)	(2106.42)	(2572.87)	(2474.12)	
R15,F01,C2	1148604	1211560	1184780	1217346.67	1192780	-2.02%
	(t)	(2593.89)	(2049.64)	(2904.63)	(2341.23)	
R15,F05,C2	2488753	2736570	2732310	2747943.33	2771770	0.87%
	(t)	(4999.33)	(5114.44)	(5778.15)	(5045.12)	
R15,F10,C2	3972667	4393650	4264090	4395146.67	4304980	-2.05%
	(t)	(6171.74)	(4594.46)	(6846.45)	(4653.12)	
R15,F01,C8	2301326.6	2355980	2350730	2360816.67	2351980	-0.37%
	(t)	(18162.70)	(14515.9)	(18763.03)	(14875.9)	
R15,F05,C8	5573412.8	5941080	5830430	5944116.67	5858273.33	-1.44%
	(16107.92)	(20972.30)	(23169.4)	(24038.43)	(22453.34)	
R15,F10,C8	8696932	9140060	9123020	9220093.33	9131563.33	-0.96%
	(18930.96)	(14527.20)	(19548.3)	(15536.40)	(18765.76)	

TAB. 4.23 – Computational Results R16 : 20,300,40

PROB	OPT	TC	PR	AV.TC	AV.PR	IMPROV
R16,F01,C1	136161	136851	136161	137553.00	136763.66	-0.57%
	(1.05)	(322.20)	(194.43)	(327.58)	(190.34)	
R16,F05,C1	239500	245003	244070	248182.00	248259.33	0.03%
	(1641.30)	(309.91)	(200.35)	(311.99)	(234.56)	
R16,F10,C1	325671	346724	330138	348585.67	342357.33	-1.79%
	(4345.92)	(306.85)	(212.21)	(312.22)	(240.34)	
R16,F01,C2	138532	140433	139131	141125.33	139996.66	-0.80%
	(27.61)	(327.39)	(184.9)	(328.64)	(203.54)	
R16,F05,C2	241801	250332	248717	253838.67	254805.66	0.38%
	(851.99)	(325.14)	(204.65)	(329.91)	(198.45)	
R16,F10,C2	337762	352450	343165	357835.33	357759	-0.02%
	(8173.47)	(305.98)	(206.89)	(315.81)	(250.45)	
R16,F01,C8	169502	173164	171651	174503.67	172527	-1.13%
	(t)	(382.96)	(209.24)	(387.08)	(278.45)	
R16,F05,C8	352976	374372	371129	376533.67	372825	-0.98%
	(t)	(429.56)	(244.07)	(434.95)	(290.45)	
R16,F10,C8	541626	587242	564151	590970.00	575334	-2.65%
	(t)	(450.30)	(229.92)	(456.95)	(249.45)	

TAB. 4.24 – Computational Results R17 : 20.300,100

PROB	OPT	TC	PR	AV.TC	AV.PR	IMPROV
R17,F01,C1	354138	367923	364325	370314.67	365378.66	-1.33%
	(332.10)	(790.06)	(536.69)	(810.44)	(654.34)	
R17,F05,C1	651025	693482	669935	696950.33	687806.33	-1.31%
	(t)	(822.09)	(710.92)	(866.93)	(806.45)	
R17,F10,C1	917956	990967	973486	996346.00	980134.66	-1.63%
	(t)	(871.10)	(845.3)	(948.55)	(765.52)	
R17,F01,C2	370590	382868	378905	385026.33	380997	-1.05%
	(2258.82)	(866.99)	(615.05)	(899.28)	(654.9)	
R17,F05,C2	708698	761476	743173	768713.67	748048.66	-2.69%
	(t)	(988.79)	(713.2)	(1026.86)	(750.54)	
R17,F10,C2	1029242	1104380	1084280	1116650.00	1098420	-1.63%
	(t)	(1092.96)	(832.08)	(1134.30)	(965.87)	
R17,F01,C8	505310	530313	521468	532371.00	523786	-1.61%
	(t)	(1622.41)	(1026.25)	(1691.09)	(1087.65)	
R17,F05,C8	1112076	1249440	1191580	1265813.33	1209456.6	-4.45%
	(t)	(2313.01)	(1486.98)	(2774.21)	(1643.98)	
R17,F10,C8	1847334	2017720	1985220	2035663.33	2014533.3	-1.04%
	(t)	(3048.11)	(2300.72)	(3165.96)	(2654.43)	

TAB. 4.25 – Computational Results R18 : 20,300,200

PROB	OPT	TC	PR	AV.TC	AV.PR	IMPROV
R18,F01,C1	828559	866838	858043	877666.67	861656.66	-1.82%
	(t)	(2167.16)	(1926.27)	(2235.21)	(2076.54)	
R18,F05,C1	1542197	1671810	1649730	1726433.33	1712110	-0.83%
	(t)	(2609.61)	(2279.48)	(2671.03)	(2465.3)	
R18,F10,C1	2292684	2383760	2418190	2457733.33	2451633.33	-0.25%
	(t)	(2659.54)	(2167.09)	(2779.20)	(2298.9)	
R18,F01,C2	921762	980992	964367	986701.67	966850.66	-2.01%
	(t)	(2717.04)	(2209.13)	(2810.17)	(2564.34)	
R18,F05,C2	1866215	2033500	1980170	2042536.67	2018590	-1.17%
	(t)	(3865.35)	(2810.98)	(4168.55)	(3032.4)	
R18,F10,C2	2895982	2981980	2928930	3011723.33	2972860	-1.29%
	(t)	(3828.76)	(3575.27)	(3939.71)	(3487.5)	
R18,F01,C8	1485690.8	1611800	1568050	1622236.67	1576570	-2.82%
	(t)	(15115.90)	(8151.71)	(19064.10)	(10321.23)	
R18,F05,C8	4010736.7	4379870	4553560	4515690.00	4690270	3.87%
	(t)	(33140.40)	(15068.2)	(34565.93)	(18654.4)	
R18,F10,C8	6663506.4	7562480	7639160	7762600.00	7998752.33	3.04%
	(t)	(31716.60)	(26296.6)	(32354.43)	(28642.6)	

Chapitre 5

Learning Mechanisms and Local Search Heuristics for the Fixed Charge Capacitated Multicommodity Network Design

Learning Mechanisms and Local Search Heuristics for the Fixed Charge Capacitated Multicommodity Network Design

Abstract

In this paper, we propose a method based on learning mechanisms to address the fixed charge capacitated multicommodity network design problem. Learning mechanisms are applied on each solution to extract meaningful fragments to build a pattern solution. Cycle-based neighborhoods are used both to generate solutions and to move along a path leading to the pattern solution by a tabu-like local search procedure. Within this concept, the method integrates important mechanisms such as intensification and diversification. Experimental results show that the proposed algorithm is effective for large structured instances with several commodities.

Keywords : Adaptive memories, Tabu search, Fixed charge capacitated multicommodity network design, Meta-heuristics, Cycle-based neighbourhoods

Résumé

Dans cet article, nous proposons une méthode basée sur des mécanismes d'apprentissage pour le problème de synthèse de réseau multiproduits avec capacités. Des mécanismes d'apprentissage sont appliqués à chaque solution pour identifier des fragments importants pour créer une solution patron. Des mouvements basés sur des cycles sont utilisés autant pour générer des solutions que pour explorer le chemin menant à la solution patron par une procédure de recherche locale avec tabous. Dans ce concept, la méthode intègre des mécanismes importants comme l'intensification et la diversification. Les résultats obtenus indiquent que l'algorithme proposé est efficace pour résoudre les problèmes de grande taille avec plusieurs produits.

Mots-clés : Mémoires adaptatives, Méthodes de recherche avec tabous, Synthèse de réseau multiproduits avec capacités, Métaheuristiques, Voisinages par cycles

5.1 Introduction

The fixed-charge capacitated multicommodity network design problem (CMND) has various applications in the field of transportation, telecommunication and production planning (Balakrishnan, Magnanti, and Mirchandani 1997, Magnanti and Wong 1984, Minoux 1989). In these applications, multiple commodities (goods, data, people, etc.) must be routed between different points of origin and destination over a network of limited capacities. Moreover, other than the routing cost proportional to the number of units of each commodity transported over a network link, a fixed cost must be paid the first time the link is used, representing its construction (opening) or improvement costs. The objective of CMND is to identify the optimal design that is, to select the links to include in the final version of the network in order to minimize the total system cost, computed as the sum of the fixed and routing costs, while satisfying the demand for transportation.

The fixed-charge capacitated multicommodity network design problem is one of the most difficult NP-hard combinatorial optimization problems. Existing exact algorithms are not yet capable to handle problems of realistic sizes (Crainic, Frangioni, and Gendron 2001, Gendron, Crainic, and Frangioni 1998, Holmberg and Yuan 2000, Sellmann, Kliewer et Koberstein 2002). Therefore, there is substantial interest in developing heuristic procedures for this problem (Crainic, Gendreau, and Farvolden 2000, Crainic, Gendron and Hernu 2002). Currently, the best available heuristic procedures are the cycle-based tabu search and the path relinking algorithms developed by Ghamlouche, Crainic, and Gendreau (2003,2004). In the first paper, the authors propose a new class of neighbourhood structures for the CMND and evaluate these neighbourhoods using a very simple tabu-based local search procedure. The approach appears robust in terms of solution quality and computing efficiency. However, it does not go beyond a rather local exploration of the search space.

Adaptive memories appear as important building blocks for designing a complete tabu search (Glover and Laguna 1997, Glover 1996, Glover, Taillard and de Werra 1993). Adaptive memories may be explicit or attributive. Explicit memory records complete solutions, typically consisting of elite solutions visited during the search

while attributive memory records information about solution attributes that change in moving from one solution to another. Ghamlouche, Crainic and Gendreau (2004) developed a path relinking method based on cycle-based tabu search that offers the best current performance among approximate solution methods for the CMND. The method makes use of explicit memory to record elite solutions. Then the process explores paths between elite solutions in order to generate improved new ones.

The motivation of this paper is to investigate effects of adding learning mechanisms to the cycle-based tabu search introduced in (2003). We aim in particular, to develop more general guidelines for the neighborhood exploration by focusing on attributive memories. Our main contribution is the adaptation to the fixed-charge capacitated multicommodity network design problem, of concepts widely used in Tabu search, such as intensification and diversification mechanisms.

The outline of the paper is as follows. Section 5.2 describes the problem then Section 5.3 provides the necessary background and fundamentals. Section 5.4 details the implementation of our learning mechanisms. Section 5.5 is dedicated to experimental results. We conclude in Section 5.6.

5.2 Problem Formulation and Notation

The goal of a CMND formulation is to find the optimal configuration - the links to include in the final design - of a network of limited capacity to satisfy the demand of transportation of different commodities sharing the network. The objective is to minimize the total system cost, computed as the sum of the link fixed and routing costs.

Given a set of commodities $\mathcal{P}$, the CMND can be defined on a network $\mathcal{G} = (\mathcal{N}, \mathcal{A})$ where $\mathcal{N}$ is the set of nodes and $\mathcal{A}$ is the set of directed arcs. A cost c_{ij}^p is associated to each unit flow of commodity p on arc (i, j), and a fixed cost f_{ij} has to be paid in order to use arc (i, j) at all. Without loss of generality, we assume that each commodity p has a single origin $o(p)$, a single destination $s(p)$, and a flow requirement of w^p units between its origin and destination nodes. The arc-based formulation of the CMND

can then be written as

$$\min z(x,y) = \sum_{(i,j)\in\mathcal{A}} f_{ij}y_{ij} + \sum_{p\in\mathcal{P}}\sum_{(i,j)\in\mathcal{A}} c_{ij}^p x_{ij}^p \tag{5.1}$$

$$\text{subject to} \sum_{j\in\mathcal{N}^+(i)} x_{ij}^p - \sum_{j\in\mathcal{N}^-(i)} x_{ji}^p = d_i^p \quad \forall i\in\mathcal{N}, \forall\, p\in\mathcal{P}, \tag{5.2}$$

$$\sum_{p\in\mathcal{P}} x_{ij}^p \leq u_{ij}y_{ij} \quad \forall (i,j)\in\mathcal{A}, \tag{5.3}$$

$$x_{ij}^p \geq 0 \quad \forall (i,j)\in\mathcal{A}, \forall p\in\mathcal{P}, \tag{5.4}$$

$$y_{ij} \in \{0,1\} \quad \forall (i,j)\in\mathcal{A} \tag{5.5}$$

where y_{ij}, $(i,j)\in\mathcal{A}$, represent the design variables that equal 1 if arc (i,j) is selected in the final design (and 0 otherwise), x_{ij}^p stand for the flow distribution decision variables indicating the amount of flow of commodity $p\in\mathcal{P}$ on arc (i,j), and

$\mathcal{N}^+(i)/\mathcal{N}^-(i)$: Set of outward/inward neighbours of node i,
u_{ij} : Capacity applied on arc (i,j),

$$d_i^p = \begin{cases} w^p & if \quad i = o(p) \\ -w^p & if \quad i = s(p) \\ 0 & \text{otherwise.} \end{cases}$$

The objective function (5.1) accounts for the total system cost, the fixed cost of arcs included in a given design plus the cost of routing the product demand, and aims to select the minimum cost design. Constraints (5.2) represent the network flow conservation relations, while constraints (5.3) state that for each arc, the total flow of all commodities cannot exceed its capacity if the arc is opened ($y_{ij}=1$) and must be 0 if the arc is closed ($y_{ij}=0$). Relations (5.5) and (5.4) are the usual non-negativity and integrality constraints for decision variables.

Recall that, for a given design vector $\bar{y}$, the arc-based formulation of the CMND becomes a capacitated multicommodity minimum cost flow problem (CMCF)

$$\min z(x(\bar{y})) = \sum_{p\in\mathcal{P}}\sum_{(i,j)\in\mathcal{A}(\bar{y})} c_{ij}^p x_{ij}^p \tag{5.6}$$

subject to (5.2) and

$$\sum_{p \in \mathcal{P}} x_{ij}^p \leq u_{ij}\bar{y}_{ij} \quad \forall (i,j) \in \mathcal{A}(\bar{y}),$$

$$x_{ij}^p \geq 0 \quad \forall (i,j) \in \mathcal{A}(\bar{y}), \forall p \in \mathcal{P},$$

where $\mathcal{A}(\bar{y})$ stands for the set of arcs corresponding to the design $\bar{y}$. A solution to the CMND may then be viewed as an assignment $\bar{y}$ of 0 or 1 to each design variable, plus the optimal flow of the corresponding multicommodity minimum cost flow problem $x^*(\bar{y})$. Similarly, the objective function value associated to a solution $(\bar{y}, x^*(\bar{y}))$ is the sum of the fixed cost of the open arcs in $\bar{y}$ and the objective function value of the CMCF associated to $x^*(\bar{y})$

$$z(\bar{y}, x^*(\bar{y})) = \sum_{(i,j) \in \mathcal{A}(\bar{y})} f_{ij}\bar{y}_{ij} + z(x^*(\bar{y})). \tag{5.7}$$

5.3 Background and Fundamentals

The necessary background of the cycle-based tabu search is outlined here. For more details, see (Ghamlouche, Crainic and Gendreau 2003). The class of neighbourhood structures proposed by Ghamlouche, Crainic and Gendreau (2003) for the CMND explores the space of the arc design variables by re-directing flow around cycles and closing and opening design arcs accordingly. The neighbourhood defines moves that explicitly take into account the impact on the total design cost of potential modifications to the flow distribution of several commodities simultaneously.

The fundamental idea is that one may move from one solution to another by 1) identifying two points in the network together with two paths connecting these points, thus closing a cycle; 2) deviating the total flow from one path to another such that at least one currently open arc becomes empty; 3) closing all previously open arcs in the cycle that are empty following the flow deviation and, symmetrically, opening all previously closed arcs that now have flow. Such neighbourhoods are huge, however, and their explicit and exhaustive exploration is not practical in most situations. Moreover, the complete evaluation of any design modification involves the resolution of a

capacitated multicommodity network flow problem, which rapidly becomes extremely computation intensive. Thus, in order to select the best move out of a given solution, the method implements an efficient procedure that 1) avoids the complete evaluation of every examined move and 2) generates a limited number of cycles that include the "good" moves. Note that not all cycles are of equal interest. The method seeks moves that modify the status of several arcs and that lead to a significant modification of the flow distribution. Therefore, moves that close at least one arc and open new paths for a group of commodities appear attractive. To close an arc, one must be able to deviate all its flow. The residual capacity of any cycle that includes that arc must then be at least equal to the total flow on the arc. Consequently, the cycles of interest are those that display a residual capacity equal to one of the values in the set of the total (strictly positive) volumes on the open arcs.

Cycles are thus to be identified on residual networks and the one leading to the network modification that yields the largest improvement (smallest deterioration, eventually) in the design objective function corresponds to the best move. To reduce the computational burden, cycles are identified and evaluated for a set of candidate links $\mathcal{C}$. The "lowest" cost cycle for each candidate link is identified by an optimization heuristic based on a modification of the shortest path label-correcting algorithm that avoids getting trapped in negative directed cycles. The method thus progressively builds a set of good candidate neighbours (cycles) among which the best move is then selected.

To evaluate these concepts, Ghamlouche, Crainic, and Gendreau (2003) developed a simple tabu search-based local search procedure that integrates two versions of the cycle-based neighbourhood : One that considers the flow of all commodities when determining cycles, and a second one that refines the search by implementing moves resulting from the deviation of the flow of only one commodity at a time.

Following an initialization phase, the tabu search procedure explores the design variables solution space using a simple local search framework : at each iteration, the best non-tabu move is determined and implemented regardless whether it improves the overall solution or not. A short-term tabu memory is used to record characteristics of visited solutions to avoid cycling. When a particularly good solution is en-

countered, the search is intensified using a particular implementation of cycle-based neighbourhoods that consider the flow distribution of one commodity only. A solution is considered particularly good when it improves the best overall solution or is close to it by at least a pre-defined percentage. The method terminates whenever a predefined stopping criterion (number of iterations, CPU time, etc.) is met. Computational results on a large set of instances, with various characteristics, show that the cycle-based tabu search produces superior solutions.

Ghamlouche, Crainic and Gendreau (2004) explore the adaptation of path relinking to the (CMND). Path relinking (Glover 1997; see also Glover and Laguna 1997 and Glover, Laguna, and Marti 2000), is a meta-heuristic that operates on a set of elite solutions, called the reference set, and generates paths between solutions in this set to create improved new ones. Starting from an "initial" solution, the primary goal of the search is to find a path to reach another, "guiding" solution, by performing moves that progressively introduce into the current solution attributes contained in the guiding solution. Thus, the method does not progress by choosing a "best" move from the neighborhood set, but by selecting the "best" move from the restricted set of moves that incorporate some or all of the attributes of the guiding solution. This exploration allows the search to perform moves that may be considered unattractive according to the objective function value but which appear essential in reaching solutions with given characteristics.

To implement path relinking, Ghamlouche, Crainic and Gendreau (2004) used the cycle-based neighbourhoods both to move along a path between elite solutions and to generate the elite candidate set by a tabu-like local search procedure. The authors proposed and compared several strategies to build the reference set and to select initial and guiding solutions from this set. The best strategies for the fixed charge capacitated multicommodity network design problem is to build the reference set with improving local minima, that is local minimum solutions that offer a better evaluation of the objective function than those already in the reference set, and to build paths between the most distant solutions in this set, that is with solutions having the maximum Hamming distance.

Extensive computational experiments indicate that the path relinking procedure

offers excellent results. It systematically outperforms the cycle-based tabu search method in both solution quality and computational effort and offers the best current meta-heuristic for this difficult class of problems.

5.4 Learning Mechanisms

During the cycle-based tabu search, each solution found depends only on the previous one. To take into consideration the history of the search, we decided to modify the cycle-based tabu search by adding learning mechanisms performed at each iteration and use this knowledge to build further solutions. To this end, we use adaptive memories that give us an overview on each arc of the network. For each arc, two adaptive memories *ArcToOpen* and *ArcToClose* are used for intensification purposes and one adaptive memory, *ArcResidency*, is used for diversification. Intensification adaptive memories record how many times it was useful to have the arc opened and how many times it was not. In particular, $ArcToOpen(i,j)$ indicates the number of times arc (i,j) is useful to be opened while $ArcToClose(i,j)$ indicates the number of times arc (i,j) is useful to be closed. The third memory, $ArcResidency(i,j)$, stores the number of times arc (i,j) has been used in a solution. *ArcResidency* is later used to diversify the search by penalizing highly used arcs and favor not much used arcs (see section 5.4.6).

For each solution, to decide whether it is good or not to open each arc, we investigate two strategies : both study the contribution of the arc to the solution cost. However in the first one, identified as arc strategy, we evaluate each arc independently while in the second, identified as node strategy, we evaluate each arc within a subset of arcs having the same origin node.

5.4.1 Arc Strategy

Two measures are used to evaluate the status of arcs of each solution of the modified cycle-based tabu search : the flow and the fixed cost. Ideally, we would like to use arcs with low fixed cost, in addition, we would like those arcs to have a very high flow. Such arcs are the most attractive and thus one would like to open these

arcs in the next solutions. Arcs with high fixed cost and low flow are poor candidates to be included in the next solutions and one would like to close all these arcs when building further solutions. Arcs with high fixed cost and high flow cannot be rejected (closed) unless we take a closer look to the fixed cost over capacity ratio. In fact, those arcs can be distributed in two subsets : those having high fixed cost over total flow ratio and those having low fixed cost over total flow ratio. Arcs in the first set (high fixed cost, high flow and high fixed cost over total flow ratio) lead to a costly objective value and should be closed in the next solutions. On the other hand, arcs in the second set (high fixed cost, high flow and low fixed cost over total flow ratio) are less expensive than those in the first set but we still do not like to open them in the next solutions because of the high fixed cost. The same argument applies on arcs having low fixed cost and low flow. Section 5.4.3 gives an evaluation mechanism to determine, for each arc, whether the flow and the fixed cost are high or low. Adaptive memories are updated for each arc of the solution by incrementing *ArcToOpen* or *ArcToClose* depending on how we would like to have the status of the arc in the next solutions. Table 5.1 summarizes the arc strategy.

ARCS	HIGH FLOW	LOW FLOW
HIGH FIXED COST	Increment *ArcToClose* for arcs having the highest fixed cost over total flow ratio	Increment *ArcToClose* for all used arcs
LOW FIXED COST	Increment *ArcToOpen* for all used arcs	Increment *ArcToOpen* for arcs having the lowest fixed cost over total flow ratio

TAB. 5.1 – Arc Strategy

In order to identify at each solution $(\bar{x}, \bar{y})$ arcs with the highest or lowest fixed cost over total flow ratio, let H be the set of all arcs having high fixed cost and high flow and L be the set of all arcs having low fixed cost and positive low flow. An arc

(i,j) belonging to H is considered to have a high fixed cost over its total flow ratio if this ratio exceeds the threshold t_1 (i.e. $f_{ij}\bar{y}_{ij}/\sum_{p\in\mathcal{P}}\bar{x}^p_{ij} > t_1$) where :

$$t_1 = \sum_{(i,j)\in H}(f_{ij}\bar{y}_{ij}/\sum_{p\in\mathcal{P}}\bar{x}^p_{ij})/|H|.$$

In the same way, an arc (i,j) belonging to L is considered to have a low fixed cost over its total flow ratio if this ratio is less than t_2 (i.e $f_{ij}\bar{y}_{ij}/\sum_{p\in\mathcal{P}}\bar{x}^p_{ij} < t_2$), where :

$$t_2 = \sum_{(i,j)\in L}(f_{ij}\bar{y}_{ij}/\sum_{p\in\mathcal{P}}\bar{x}^p_{ij})/|L|.$$

5.4.2 Node Strategy

In another attempt to use flow and fixed cost to build adaptive memories, information on arcs are grouped and transfered to their originated nodes as follow :

F_i = The total fixed cost on outgoing arcs from node i having positive flow. $(\sum_{(i,j)\in\mathcal{A}/j\in i+} f_{ij}\bar{y}_{ij})$

X_i = The total flow on outgoing arcs from node i. $(\sum_{p\in\mathcal{P}}\sum_{(i,j)\in\mathcal{A}/j\in i+}\bar{x}^p_{ij})$

where $(\bar{x},\bar{y})$ denote the current solution and j is the node successor of i. Section 5.4.3 gives an evaluation mechanism to determine, for each node, whether the flow and the fixed cost are high or low.

Transferring information from arcs to nodes is based on the fact that flow is traveling on arcs, flow is then grouped on nodes and then redistributed on arcs. Consequently, at each node, the cost of this redistribution is the total cost on outgoing arcs. In node strategy, adaptive memories are updated according to the influence of this redistribution on the objective value as follows :

- If node i has a high fixed cost and a low flow, the distribution of the flow is very costly and we want to avoid this redistribution in the next solutions. In fact, we want to push the flow back to use different distribution channels. This can be realized by closing all used arcs originating from node i and thus *ArcToClose* will be incremented for those arcs.

- If node i has a low fixed cost and a high flow, the distribution cost of the flow is very low, thus we want to favor this distribution in next solutions. This can be realized by keeping all used arcs originating from node i opened. Consequently, $ArcToOpen$ will be incremented for those arcs.

- If node i has a high fixed cost and a high flow, the distribution cost is more or less acceptable. In this case, we might need to tighten the flow by closing some arcs. In fact we are interested in preventing part of the flow (i.e. causing high redistribution cost) to be redistributed on outgoing arcs in the next solutions. To identify the part of the flow to be pushed back, let H_i be the set of used arcs originating from node i. An arc (i,j) belonging to H_i should be closed ($ArcToClose$ incremented) if its fixed cost over total flow ratio exceeds the threshold t_{ci}, computed as the average fixed cost over total flow of all used arcs originating at node i. Explicitly, $ArcToClose(i,j)$ is incremented if :

$$(f_{ij}\bar{y}_{ij}/\sum_{p\in\mathcal{P}}\bar{x}_{ij}^{p}) > t_{ci} = \sum_{(i,j)\in H_i}(f_{ij}\bar{y}_{ij}/\sum_{p\in\mathcal{P}}\bar{x}_{ij}^{p})/|H_i|.$$

- If node i has a low fixed cost and a low flow, the distribution cost is also more or less acceptable. In fact, only the part of the flow leading to the lowest redistribution cost is attractive and we would like to keep this part redistributed on outgoing arcs in the next solutions. To identify the part of the flow to be pushed forward, let L_i be the set of used arcs originated at node i. An arc (i,j) belonging to L_i should be kept opened ($ArcToOpen$ incremented) if its fixed cost over total flow ratio is lower than a threshold t_{oi}, computed as the average fixed cost over total flow of all used arcs originated at node i. Explicitly, $ArcToOpen(i,j)$ is incremented if :

$$(f_{ij}\bar{y}_{ij}/\sum_{p\in\mathcal{P}}\bar{x}_{ij}^{p}) < t_{oi} = \sum_{(i,j)\in L_i}(f_{ij}\bar{y}_{ij}/\sum_{p\in\mathcal{P}}\bar{x}_{ij}^{p})/|L_i|.$$

5.4.3 Evaluation Mechanisms

Two thresholds are used to explicitly identify low/high fixed cost and low/high flow on arcs (or nodes) at each solution of the modified cycle-based tabu search. These thresholds depend on the current best solution found and thus are tighter when progresses the search. Those thresholds are calculated as follows :

Let x and f denote respectively the percentage of used capacity and the average used fixed cost of the current best solution $(\tilde{x}, \tilde{y})$:

$$x = \frac{\sum_{p \in \mathcal{P}} \sum_{(i,j) \in A} \tilde{x}_{ij}^{p}}{\sum_{(i,j) \in \mathcal{A}} u_{ij} \tilde{y}_{ij}}$$

$$f = \frac{\sum_{(i,j) \in \mathcal{A}} f_{ij} \tilde{y}_{ij}}{\sum_{(i,j) \in \mathcal{A}} \tilde{y}_{ij}}$$

In *arc* strategy, an arc (i, j) of the current solution $(\bar{x}, \bar{y})$ has a high fixed cost if its used fixed cost exceeds f $(f_{ij}\bar{y}_{ij} > f)$ otherwise the arc has a low fixed cost $(f_{ij}\bar{y}_{ij} \leq f)$. In the same way, arc (i, j) has a high flow if its total flow over total capacity ratio exceeds x $(\sum_{p \in \mathcal{P}} \bar{x}_{ij}^{p} / u_{ij} > x)$ otherwise the arc has a low flow.

Following the same analogy with *node* strategy, a node i of the current solution $(\bar{x}, \bar{y})$ has a high fixed cost if the average fixed cost on outgoing arcs from node i exceeds f. Explicitly :

$$\frac{F_i}{\sum_{(i,j) \in \mathcal{A} / j \in i^+} \bar{y}_{ij}} > f$$

In the same way, node i has a high flow if the percentage of used capacity on outgoing arcs from node i exceeds x. Explicitly :

$$\frac{X_i}{\sum_{(i,j) \in \mathcal{A} / j \in i^+} u_{ij} \bar{y}_{ij}} > x$$

5.4.4 Pattern solution

To move from a current solution to a neighboring one in our approach, we select a set of candidate arcs from the network and we perform cycle-based tabu search to

get the best move. Remember that at least one of the candidate arcs will have its status changed after the move. In the original form of the cycle-based tabu search, the candidate arcs were selected randomly from the set of closed arcs. In our approach, we will use the learning mechanisms to decide which arcs should be included in the candidate set. Each closed arc is a candidate arc if *ArcToOpen* exceeds *ArcToClose* by a predefined value. This means that during the search, it was more useful to have the arc opened than to have it closed. However, since the arc is closed in the current solution, we want to direct the search to open the arc without forcing it to be opened. Similarly, each opened arc is a candidate arc if *ArcToClose* exceeds *ArcToOpen* by a predefined value.

Pattern solution is the result of our learning during the search. In *pattern* solution, an arc is opened if its associated *ArcToOpen* exceeds *ArcToClose* by a predefined value *OpenTheArc* and an arc is closed if its associated *ArcToClose* exceeds *ArcToOpen* by a predefined value *CloseTheArc*. Originally, all arcs in *pattern* solution have an undecided status. In this way, arcs with different status between current and *pattern* solution constitute the set of candidate arcs when performing a cycle-based tabu search.

5.4.5 Intensification

Building the set of candidate arcs as in section 5.4.4 constitute an intensification in the already explored neighborhood since we seek, via the *pattern* solution, to open arcs found good during the search and to close those arcs found to be costly. The intensification phase thus consists on using the cycle-based tabu search to find the best move starting from the current solution and building and maintaining the *pattern* solution as in section 5.4.4. The intensification phase ends after a given number of iterations, $MaxInt$, without an improvement of the objective function value.

Note that, even if adaptive memories are updated after each iteration of the search, the *pattern* solution is only updated when no improvement to the current solution is noticed. This is to avoid disturbing the search when improvement is taking place.

5.4.6 Diversification

Intensification by itself is insufficient to yield the best outcome to our difficult problem. Diversification must be invoked to allow the most effective search over the solution space. To do this, we exploit additional memory means (i.e residency based memories) to penalize frequently occurred arcs in visited solutions and consequently reach a new search trajectory over the solution space. As in the intensification phase, we use the *pattern* solution to build the set of candidate arcs and the cycle-based tabu search to move from one solution to another. However, the *pattern* solution is modified to introduce the residency based memory as follow : An arc (i, j) is set to be opened in *pattern* solution if its current status is not decided and its $ArcResidency$ is less than a predefined threshold $ResidMeasure$. An arc (i, j) is set to be closed in *pattern* solution if its current status is not decided and its $ArcResidency$ exceeds a $ResidMeasure$. All other arcs of the *pattern* solution will receive a not decided status. The diversification is launched after the intensification phase and performed for $MaxDiv$ iterations without improvement of the best solution.

5.4.7 Path relinking

In our approach, path relinking is implemented to explore trajectories connecting best solutions found during the intensification and the diversification phases. In particular, when moving toward *pattern* solution in the intensification phase, we keep track of the best r solutions to build a first reference set, then when the method switches to a diversification phase, a second reference set is built with another r best solutions. The method then starts to explore trajectories connecting those solutions : at each iteration, two solutions, one from each reference set, that satisfy the maximum Hamming distance are chosen, the worst one is set to be the starting solution and is removed from its reference set. If during the path relinking exploration, we reach a solution that improves the best overall solution, this solution is added to the reference set of the starting solution. If one of the reference sets is empty, path relinking keeps exploring trajectories between solutions in the remaining reference set. Path relinking ends when both reference sets are empty.

5.4.8 Warming Up

The method needs time to learn and a way to start the search. Therefore a warming up phase is performed. It consists on identifying arcs to open or to close and applying cycle-based tabu search on these arcs to get the best move. At each iteration of the warming up phase, adaptive memories are updated and used to build *pattern* solution. However, at this stage, *pattern* solution is not yet mature to guide the search in the solution space and will be used only at the end of the warming up phase. Two warming up phases are performed : the first one consists in closing arcs with high fixed cost over total flow ratio while the second consists in opening arcs with low fixed cost over capacity ratio. The percentage of arcs to be closed or opened is set to 50% of the total number of opened or closed arcs. This value was selected as the best during the experimental results reported in Ghamlouche, Crainic and Gendreau (2003). The warming up stops after a given number of iterations, $MaxWarmingUp$, without an improvement in the objective function value.

5.4.9 The Search Strategy

After some initialization, the method performs a warming up phase to create *pattern* solution. When no improvement is observed, the search proceeds to an intensification phase until a number of iterations without improving the best overall solution is reached. The method switches then to a diversification phase as indicated in Section 5.4.6. Path relinking is then applied to explore paths connecting best solutions found during the intensification and the diversification phases. The overall process is then repeated by starting with the best overall solution. Figure 5.1 summarized the general structure of the search while Figure 5.2 details the learning phase.

5.5 Experimentation and Computational Results

Experiments have been performed to evaluate the behavior and the performance of the learning algorithm proposed in this paper. To ensure meaningful comparisons, we employ the same two sets of problem instances as used in Ghamlouche, Crainic, and Gendreau (2003,2004).

Initialization

Generate an initial feasible solution to initiate *BestSolution* and *CurrentSolution*
Let $(\tilde{x}, \tilde{y})$, $(\bar{x}, \bar{y})$ and (x^t, y^t) denote the *BestSolution*, *CurrentSolution* and *PatternSolution* respectively

Set WarmingUpStatus = closed

Main search loop

Repeat the following until a stopping condition is met

Initialize memories
Repeat until MaxWarmingUp is reached

- if (WarmingUpStatus = opened)
  - Sort closed arcs of the current solution according to f_{ij}/u_{ij}
  - Let $\Gamma = \{(i,j)/\bar{y}_{ij} = \textit{closed}$ and f_{ij}/u_{ij} is low $\}$
- else if (WarmingUpStatus = closed)
  - Sort used arcs of the current solution according to $f_{ij}/\sum_{p\in\mathcal{P}} \bar{x}^p_{ij}$
  - Let $\Gamma = \{(i,j)/\bar{y}_{ij} = \textit{opened}$ and $f_{ij}/\sum_{p\in\mathcal{P}} \bar{x}_{ij}$ is high $\}$
- Perform one iteration of cycle-based neighbourhood by considering arcs in Γ to get a new current solution $(\bar{x}, \bar{y})$
- Perform a Learning phase
- If *CurrentSolution* < *BestSolution* update *BestSolution*

- **Intensification Loop**

  Repeat until $MaxInt$ is reached

  - Perform one tabu search iteration by introducing arcs present in *Pattern* solution to get a new solution $(\bar{x}, \bar{y})$
  - Perform a Learning phase
  - Save best solutions in the first reference set

- **Perform a Diversification phase by changing *Pattern* solution**

  Repeat until $MaxDiv$ is reached

  - Perform one tabu search iteration by introducing arcs present in *Pattern* solution to get a new solution $(\bar{x}, \bar{y})$
  - Perform a Learning phase
  - Save best solutions in the second reference set

- Perform a Path Relinking phase between best solutions found during both Intensification and diversification phases
- Set *CurrentSolution* to *BestSolution*
- WarmingUpStatus = opened

FIG. 5.1 – Global Procedure

- for each arc (i,j) of *CurrentSolution* $(\bar{x}, \bar{y})$ with $\bar{y}_{ij} = 1$ increment ArcResid(i,j)
- Calculate x and f as in section 5.4
- **if arc strategy**

  Low fixed cost, high flow

  - If $f_{ij}\bar{y}_{ij} < f$ and $\sum_{p \in P} \bar{x}^p_{ij}/u_{ij} > x$ then increment ArcToOpen(i,j)

  High fixed cost, low flow

  - If $f_{ij}\bar{y}_{ij} > f$ and $\sum_{p \in P} \bar{x}^p_{ij}/u_{ij} < x$ then increment ArcToClose(i,j)

  High fixed cost, high flow

  - Calculate the threshold t_1
  - Increment *ArcToClose* for each arc having $f_{ij}\bar{y}_{ij} / \sum_{p \in P} \bar{x}^p_{ij} > t_1$

  Low fixed cost, low flow

  - Calculate the threshold t_2
  - Increment *ArcToOpen* for each arc having $f_{ij}\bar{y}_{ij} / \sum_{p \in P} \bar{x}^p_{ij} < t_2$
- **if node strategy**
  - for each node i of *CurrentSolution* $(\bar{x}, \bar{y})$
    - Calculate $fixedcost(i)$ and $flow(i)$ as in section 5.4.2

    Low fixed cost, high flow

    - If $fixedcost(i) < f$ and $flow(i) > x$ then
      - for each arc (i,j) of *CurrentSolution* $(\bar{x}, \bar{y})$ with $\bar{y}_{ij} = 1$ and $j \in i^+$ increment ArcToOpen(i,j)

    High fixed cost, low flow

    - if $fixedcost(i) > f$ and $flow(i) < x$ then
      - for each arc (i,j) of *CurrentSolution* $(\bar{x}, \bar{y})$ with $\bar{y}_{ij} = 1$ and $j \in i^+$ increment ArcToClose(i,j)

    High fixed cost, high flow

    - if $fixedcost(i) > f$ and $flow(i) > x$ then
      - Calculate the threshold t_{ci}

        Increment *ArcToClose* for each arc outgoing having $f_{ij}\bar{y}_{ij} / \sum_{p \in P} \bar{x}^p_{ij} > t_{ci}$

    Low fixed cost, low flow

    - if $fixedcost(i) < f$ and $flow(i) < x$ then
      - Calculate the threshold t_{oi}

        Increment *ArcToOpen* for each outgoing arc having $f_{ij}\bar{y}_{ij} / \sum_{p \in P} \bar{x}^p_{ij} < t_{oi}$
  - if (Warming up) or $(CurrentSolution \geq PreviousSolution)$
    - for each arc $(i,j) \in PatternSolution$ do
      - if ArcToOpen(i,j) - ArcToClose(i,j) $\geq OpenTheArc$ then y^t_{ij} = Opened
      - if ArcToClose(i,j) - ArcToopen(i,j) $\geq CloseTheArc$ then y^t_{ij} = Closed

FIG. 5.2 – Learning phase

The heuristic in this paper were implemented in C++. The exact evaluation of the capacitated multicommodity network flow problems is done using the linear programming solver of CPLEX 7.5. (2002). All tests were conducted on one 400MHz processor of a 64-processors Sun Enterprise 10000 with 64 Gigabyte of RAM, operating under Solaris 2.7. Computing times are reported in seconds.

5.5.1 Parameter settings

We first performed a calibration phase. An initial set of results (not shown here in order not to overcharge the paper) allowed us to fix the value of $MaxWarmingUp$, $MaxInt$ and $MaxDiv$ to 10, 40 and 40 respectively. Ten problems have been selected for calibration purposes. The ten problems cover networks sizes from 100 to 700 design arcs and from 10 to 400 commodities. They also display relatively high fixed cost compared to routing cost and are tightly capacitated. We tested the following combinations of parameters :

- *OpenTheArc* : This parameter indicates the threshold to exceed in order to open the arc in *Pattern* solution.
- *CloseTheArc* : This parameter indicates the threshold to exceed in order to close the arc in *Pattern* solution.

Three values 1, 2 and 3 were considered initially for these parameters. However, the value of 1 for *OpenTheArc* and *CloseTheArc* and consequently all combinations $(1,1), (1,2), (1,3)$ as well as $(2,1)$ and $(3,1)$, were rapidly dropped since the quality of the solutions started to decrease.

- *ResidMeasure* : This parameter depends on the number of iterations and indicates how often the arc should be opened to be considered highly used. Three values 40%, 60% and 80% of the number of current iteration have been tested.

Node strategy was used for these tests. Each parameter combination was ranked for each problem instance according to the gap relative to the best known solution (that of the branch-and-bound procedure of CPLEX 7.5, when available, or that obtained by Ghamlouche, Crainic and Gendreau 2004, otherwise). A score of 10, .., 1 is assigned to each of the first ten places, respectively. The performance of each parameter setting is then aggregated : gaps are averaged while scores are summed up.

Table 5.2 displays these aggregated results for each parameter combination. The first column holds the parameter setting, the second column presents the global average gaps while the last column displays the total score.

TAB. 5.2 – Parameter Setting Performances

Parameter Setting	Gap	Score
2, 2, 40%	2.35%	59
2, 2, 60%	2.67%	40
2, 2, 80%	2.80%	41
2, 3, 40%	2.36%	33
2, 3, 60%	1.62%	72
2, 3, 80%	1.85%	67
3, 2, 40%	2.16%	69
3, 2, 60%	2.44%	35
3, 2, 80%	2.03%	51
3, 3, 40%	2.42%	63
3, 3, 60%	2.37%	24
3, 3, 80%	2.80%	34

The results in Table 5.2 display one set of parameters that offers the most robust combination, $OpenTheArc = 2$, $CloseTheArc = 3$ and $ResidMeasure = 60\%$. It offers the lowest average gap and the highest scores. This setting will be maintained in the remaining of this computational study.

5.5.2 Performance Analysis

To evaluate the behavior and the performance of the learning algorithm proposed in this paper, we compare its output to the results of the cycle-based tabu search and to those of the path relinking algorithm (2003,2004). To further characterize the quality of the solutions, we also include the optimal solutions obtained using the branch-and-bound algorithm of CPLEX 7.5 (2002).

The same two data sets of networks used by Ghamlouche, Crainic, and Gendreau (2003) were also used to test our learning algorithm. Problems in both sets are general transshipment networks with no parallel arcs. Each commodity corresponds to a single origin-destination pair. On each arc, routing costs are the same for all commodities.

Problem instances have been generated to offer for each network size a variety of fixed cost to routing cost ratios and capacity to demand ratios. Detailed description of problem instances is given in Crainic, Frangioni, and Gendron (2001); see also Gendron and Crainic (1994, 1996). The problem generators as well as the problem instances can be obtained from the authors.

Problems in the first set of network, denoted **C**, are defined respectively by the number of nodes, the number of arcs, the number of commodities as well as two letters summarizing the fixed cost and capacity information : a relatively high or low fixed cost relative to the routing cost is signaled by the letter **F** or **V**, respectively, while letters **T** and **L** indicate respectively if the problem is tightly or somewhat loosely capacitated compared to the total demand.

Computational results for the first set of networks are reported in Tables 5.3, 5.4 and 5.5. In these tables, the OPT column corresponds to the solution of the branch-and-bound algorithm solved using CPLEX 7.5 (2002) on the same workstations. A limit of 10 hours was imposed. An **X** indicates that the procedure has failed to produce a feasible solution within this time limit, while a **t** indicates that the procedure stopped due to a time limit condition. The columns labeled TC and PR hold respectively the best solution, over 3 runs, of the cycle-based tabu search and the path relinking approach while AV.TC and AV.PR columns display respectively the average solution found by these two meta-heuristics. The column LS-NODE and LS-ARC gives the solutions obtained by our approach when using node strategy and arc strategy respectively. When our learning algorithm produces optimal solutions or solutions better than the best solutions found by path relinking, bold characters are employed. The figures in parentheses represent total computation time in CPU seconds. For comparison purposes, gap is computed for solutions of our learning algorithm with respect to the average solution found by path relinking and displayed in percentage under the CPU time in columns LS-NODE and LS-ARC respectively.

Table 5.3 shows the results of our learning algorithm on smaller testcases (number of commodities up to 100) while Tables 5.4 and 5.5 show the results from runs of larger structured testcases (i.e with 200 and 400 commodities). From the numerical results, a number of observations can be made. First, the use of adaptive memories

is effective for realizing good quality solution for our difficult problem. The results of Table 5.4 show that our proposed algorithm improves the best solutions found by path relinking for 6 out 8 problems. Results are also encouraging for large real-world problems, such those with 400 commodities (see Tables 5.4 and 5.5). For this class of difficult instances, CPLEX is unable to find the optimal solution, (not even a feasible solution for 7 out 8 problems) with available computational time. Our learning algorithm improves the best known solutions for 6 out 8 problems and the improvement relative to the average solution of path relinking ranges from 1.60% to 3.14%.

The second observation concerns the computing effort. Even if we account for the fact that we stop our learning algorithm on the same criterion as the path relinking procedure, 400 iterations, it appears that the learning algorithm requires longer computing times, especially when the number of commodities is high. This could be explained by the fact that good solutions are found earlier in the search (of the order of 40% of the total iteration limit), which yields more difficult multicommodity capacitated network flow problems to be solved by CPLEX at each iteration and consequently more time.

The interesting performance and behavior of the proposed learning algorithm is confirmed by the results obtained on the second set of problems instances, denoted **R**. There are 116 problems divided in 18 groups. Each group contains the same number of nodes, arcs, and commodities but with different combined level of fixed cost and capacity ratios. Three levels of fixed cost and capacity ratios are considered : F01 = 0.01, F05 = 0.05, and F10 = 0.10 indicating continuously higher levels of fixed costs compared to routing costs, and C1 = 1, C2 = 2, and C8 = 8 that signal that the total capacity available becomes increasingly tighter relatively to the total demand. The fixed cost ratio is computed as $|\mathcal{P}| \sum_{(ij)\in\mathcal{A}} f_{ij} / \sum_{p\in\mathcal{P}} w^p \sum_{(ij)\in\mathcal{A}} c^p_{ij}$, and the capacity ratio is computed as $|\mathcal{A}| \sum_{p\in\mathcal{P}} w^p / \sum_{(ij)\in\mathcal{A}} u_{ij}$. Only aggregated statistics are used in the following to support the discussion. Detailed results are included in the Appendix.

Table 5.6 displays the optimality gap distribution, according to problem dimension, for both learning strategy and path relinking algorithms. As we can see, our learning algorithm achieves better results than the average solution of path relinking

PROB	OPT	TC	PR	AV. TC	AV. PR	LS-NODE	LS-ARC
25,100,10,V,L	14712 (0.36)	14712 (19.08)	14712 (12.97)	14769.33 (19.38)	14712 (13.01)	**14712** (12.36) 0%	**14712** (12.26) 0%
25,100,10,F,L	14941 (53.64)	14941 (22.55)	14941 (15.2)	14941 (22.80)	15081.33 (16.3)	**14941** (12.9) -0.93%	**14941** (13.6) -0.93%
25,100,10,F,T	49899 (40.58)	50529 (31.39)	49899 (22.1)	50619.67 (32.10)	50154 (24.5)	50324 (18.75) 0.34%	51328 (21.3) 2.34%
25,100,30,V,T	365272 (16.62)	365385 (121.30)	365322 (91.9)	365385 (123.98)	365323.66 (93.2)	365322 (83.11) -0.02%	365322 (81.81) -0.02%
25,100,30,F,T	85530 (534.18)	87325 (123.88)	86428 (97.78)	88095.33 (125.69)	86492.33 (99.67)	**86334** (89.31) -0.18%	86815.3 (84.29) 0.37%
100,400,10,V,L	28423 (84.81)	28786 (208.58)	28485 (83.97)	28836.67 (225.12)	28529 (89.9)	28553 (83.11) 0.08%	28553 (83.31) 0.08%
100,400,10,F,L	24436 (t)	24022 (178.52)	24022 (109.66)	24022 (191.07)	24022 (112.45)	24104 (77.45) 0.34%	24104 (77.5) 0.34%
100,400,10,F,T	66364 (t)	67184 (425.68)	65278 (193.4)	68215.00 (432.01)	65153 (201.34)	66171 (340.63) 0.96%	66410 (218.58) 1.33%
100,400,30,V,T	385544 (t)	385508 (1161.06)	384926 (424.08)	385512.7 (1169.70)	385181.7 (450.76)	384951 (563.7) -0.06%	**384828** (631.31) -0.09%
100,400,30,F,L	50496 (t)	51831 (730.10)	51325 (328.08)	52176.33 (648.47)	51875.67 (301.4)	53066 (322.31) 2.29%	52173 (293.71) 0.57%
100,400,30,F,T	141278 (t)	147193 (1208.95)	141359 (529.02)	147478 (1235.27)	143403.7 (579.32)	143552 (619.96) 0.10%	142411 (462.89) -0.67%
30,520,100,F,T	98357 (t)	105130 (2863.76)	106130 (1336.9)	107885.3 (2418.28)	107575 (1405.5)	106912 (1134.01) -0.62%	107266 (1257.51) -0.28%
30,700,100,F,T	55709 (t)	57628 (3219.31)	56575.5 (1534.6)	58111.33 (3316.46)	56808.83 (1765.3)	57741 (1840.27) 1.64%	58032.7 (2266.3) 2.15%

TAB. 5.3 – Computational Results, C problems

PROB	OPT	TC	PR	AV.TC	AV.PR	LS-NODE	LS-ARC
20,230,200,V,L	94386 (t)	100001 (2577.34)	100404 (2317.35)	101481.3 (2606.79)	101469.3 (2034.54)	102492 (3235.53) 1.01%	102492 (3323.53) 1.01%
20,230,200,F,L	141737.4 (t)	148066 (3143.76)	147988 (2893.49)	148975 (3158.22)	151352 (2760.73)	150617 (2955.02) -0.49%	151961 (3229.97) 0.40%
20,230,200,V,T	97914 (t)	106868 (2595.35)	104689 (2304.25)	107589.3 (2361.57)	105598.7 (2304.63)	**103700** (2921.11) -1.80%	**103700** (2934.62) -1.80%
20,230,200,F,T	137271 (t)	147212 (3601.90)	147554 (3656.96)	147868 (3868.50)	148044.3 (3505.46)	**144895** (4220.34) -2.13%	149284 (4732.58) 0.84%
30,520,400,V,L	112997.5 (t)	122673 (55771.2)	119416 (29650.7)	123277.3 (55720.30)	119624 (33716.23)	**115918** (82551.2) -3.10%	**115918** (95761.4) -3.10%
30,520,400,F,L	X (t)	164140 (429296)	163112 (33641.2)	165458 (40922.57)	163377 (35671.23)	**161205** (54686.2) -1.32%	**159084** (62757.9) -2.63%
30,520,400,V,T	X (t)	122655 (46565.2)	120170 (31461.9)	123210 (50666.83)	120764.3 (25705.4)	**118835** (44631.4) -1.60%	**118705** (60088.6) -1.70%
30,520,400,F,T	X (t)	169508 (49886.9)	163675 (51400.1)	170301.3 (49476.67)	164921.3 (44862.3)	**161102** (114120.4) -2.32%	**161102** (107664) -2.32%

TAB. 5.4 – Computational Results, **C** problems

PROB	OPT	TC	PR	AV.TC	AV.PR	LS-NODE	LS-ARC
20,300,200,V,L	74972.4 (t)	81367.7 (4247.14)	78184 (3487.2)	82187.23 (3974.91)	79095.33 (3560.5)	79791.5 (4899.92) 0.88%	81307 (4473.87) 2.80%
20,300,200,F,L	117306 (t)	122262 (3267.94)	123484 (3765.34)	123247.7 (4524.49)	124924.7 (3912.5)	128258 (4664.53) 2.67%	125421 (4046.51) 0.40%
20,300,200,V,T	74991 (t)	80344 (4235.15)	78866.8 (3691.24)	82187.23 (4396.22)	79212.27 (3860.3)	81453 (5158.17) 2.83%	81453 (4771.54) 2.83%
20,300,200,F,T	108252 (t)	113947 (4657.54)	113584 (3546.58)	115342 (4906.50)	114632.3 (4001.23)	114269 (4691.77) -0.32%	114259 (4532.14) -0.15%
30,700,400,V,L	X (t)	107727 (36332)	105116 (22314.6)	108459.3 (36282.63)	105403.3 (19733.4)	**102906** (45637) -2.36 %	**102530** (54264.3) -2.73%
30,700,400,F,L	X (t)	150256 (73030.8)	145026 (52360.2)	150909 (63170.20)	147887.3 (58761.8)	148862 (128777) 0.66%	146921 (125860) -0.65%
30,700,400,V,T	 (t)	101749 (49324.2)	101212 (26592.3)	103112.3 (49239.5)	102119.3 (28664.3)	**98911** (99322.9) -3.14%	**98911** (102727) -3.14%
30,700,400,F,T	X (t)	144852 (74796.3)	141013 (45179.1)	146705 (79053.40)	141446.7 (49824.83)	**139055** (95330.1) -1.69%	141096 (123233) -0.25%

TAB. 5.5 – Computational Results, C problems

when the number of commodities increases versus the number of arcs. The results also indicate that our learning algorithm is certainly not interesting for small-sized problem instances with few commodities. On the other hand, it helps to achieve better solutions, especially in the case of more difficult problems, with a large number of commodities.

TAB. 5.6 – Gap Distribution According to Problem Dimensions

$\|N\|, \|A\|$	$\|P\|$	PR	AV.PR	LS-NODE	LS-ARC
	10	0%	0%	0%	0%
10,25	25	0.23%	0.60%	0.43%	0.39%
	50	0.61%	0.72%	**0.49%**	**0.44%**
	10	0.08%	0.12%	0.25%	**0.04%**
10,50	25	0.36%	0.55%	0.46%	0.48%
	50	1.14%	1.93%	**1.05%**	1.15%
	10	0.04%	0.46%	0.89%	1.02%
10,75	25	0.41%	0.92%	0.72%	0.87%
	50	1.52%	2.30%	2.37%	2.40%
	40	1.37%	1.70%	3.02%	2.29%
20,100	100	2.05%	2.82%	2.22%	2.45%
	200	4.55%	4.86%	**2.81%**	**3.26%**
	40	3.59%	4.84%	6.29%	5.32%
20,200	100	4.93%	5.90%	7.79%	7.50%
	200	5.41%	6.42%	**5.00%**	**5.07%**
	40	2.08%	3.91%	6.89%	7.28%
20,300	100	4.68%	5.79%	7.96%	7.50%
	200	6.84%	8.97%	**4.28%**	**5.12%**

According to the results, we observe that our learning algorithm is well fitted to a large number of instances and its performance increases with increasing the difficulty of problems.

5.6 Conclusion

The objective of our study has been to exploit and advance the knowledge associated with the implementation of learning mechanism with the use of adaptive memories structure for the fixed charge capacitated multicommodity network design problems. In particular, we have undertaken to examine the critical issue of what form of intensification and diversification proves more effective for this class of difficult problem. Our resulting algorithm is effective, and its benefits are particularly significant for large structured instances.

References

Balakrishnan, A., Magnanti, T.L., and Mirchandani, P. (1997). Network Design. In Dell'Amico, M., Maffioli, F., and Martello, S., editors, *Annotated Bibliographies in Combinatorial Optimization*, pages 311–334. John Wiley & Sons, New York, NY.

Crainic, T.G., Frangioni, A., and Gendron, B. (2001). Bundle-Based Relaxation Methods for Multicommodity Capacitated Network Design. *Discrete Applied Mathematics*, 112 :73–99.

Crainic, T.G., Gendreau, M., and Farvolden, J.M. (2000). A Simplex-Based Tabu Search Method for Capacitated Network Design. *INFORMS Journal on Computing*, 12(3) :223–236.

Crainic, T.G, Gendron, B, and Hernu, G. (2002). A slope Scaling/Lagrangean Perturbation Heuristic with Long-Term Memory for Multicommodity Capacitated Fixed-Charge Network Design.

Gendron, B., Crainic, T.G., and Frangioni, A. (1998). Multicommodity Capacitated Network Design. In Sansó, B. and Soriano, P., editors, *Telecommunications Network Planning*, pages 1–19. Kluwer Academic Publishers, Norwell, MA.

Gendron, B. and Crainic, T.G. (1994). Relaxations for Multicommodity Network Design Problems. Publication CRT-965, Centre de recherche sur les transports, Université de Montréal, Montréal, QC, Canada.

Gendron, B. and Crainic, T.G. (1996). Bounding Procedures for Multicommodity Capacitated Network Design Problems. Publication CRT-96-06, Centre de recherche sur les transports, Université de Montréal, Montréal, QC, Canada.

Ghamlouche, I., Crainic, T.G., and Gendreau, M. (2003). Cycle-based Neighbourhoods for Fixed-Charge Capacitated Multicommodity Network Design. *Operations Research*, 51(4) :655–667.

Ghamlouche, I., Crainic, T.G., and Gendreau, M. (2004). Path Relinking, Cycle-based Neighbourhoods and Capacitated Multicommodity Network Design. *Annals of Operations Research.* to appear.

Glover, F., Laguna, M., and Marti, R. (2000). Fundamentals of scatter search and path relinking. *Control and Cybernetics*, 39(3) :653–684.

Glover, F. and Laguna, M. (1997). *Tabu Search.* Kluwer Academic Publishers, Norwell, MA.

Glover, F., Taillard, É.D., and de Werra, D. (1993). A user's guide to tabu search. *Annals of Operations Research*, 41 :3–28.

Glover, F. (1996). Tabu Search and Adaptive Memory Programming – Advances, Applications and Challenges. In Barr, R., Helgason, R., and Kennington, J., editors, *Interfaces in Computer Science and Operations Research*, pages 1–75. Kluwer Academic Publishers, Norwell, MA.

Glover, F. (1997). A Template for Scatter Search and Path Relinking. In Hao, J., Lutton, E., Ronald, E., Schoenauer, M., and Snyers, D., editors, *Artificial Evolution*, volume 1363 of *Lecture Notes in Computer Science*, pages 13–54. Springer Verlag, Berlin.

Holmberg, K. and Yuan, D. (2000). A Lagrangean Heuristic Based Branch-and-Bound Approach for the Capacitated Network Design Problem. *Operations Research*, 48(3) :461–481.

ILOG (2002). *ILOG CPLEX 7.5.* ILOG, Mountain View, CA. U.S.A.

Magnanti, T.L. and Wong, R.T. (1984). Network Design and Transportation Planning : Models and Algorithms. *Transportation Science*, 18(1) :1–55.

Minoux, M. (1989). Network Synthesis and Optimum Network Design Problems : Models, Solution Methods and Applications. *Networks*, 19 :313–360.

Sellmann, M., Kliewer, G., and Koberstein, A. (2002). Capacitated Network Design, Cardinality Cuts and Coupled Variable Fixing Algorithms based on Lagrangian Relaxations. Publication tr-ri-02-234, University of Paderborn, Department of Mathematics and Computer Science.

Appendix

Results for each problem instance of set **R** are given in Tables 4.8 to 4.25. The following information is displayed :

- OPT : The optimal solution obtained by using the branch-and-bound algorithm of CPLEX version 7.5. A (t) indicates that the procedure stopped due to a time limit condition. A limit of 10 hours is imposed.
- TC and PR : The best solution obtained respectively by the cycle-based tabu search and path relinking over 3 run.
- AV.TC and AV.PR : The average solution obtained respectively by the cycle-based tabu search and path relinking over 3 run.
- LS-NODE : gives the solutions obtained by our approach when using node strategy.
- LS-ARC : gives the solutions obtained by our approach when using arc strategy.
- All tests were conducted on one 400MHz processor of a 64-processors Sun Enterprise 10000 with 64 Gigabyte of RAM, operating under Solaris 2.7. The figures in parentheses in columns OPT, TC, PR, LS-NODE and LS-ARC represent total computation time in CPU while these figures in columns AV.TC and AV.PR represent average total computation time.

PROB	OPT	TC	PR	AV.TC	AV.PR	LS-NODE	LS-ARC
R01,F01,C1	74079 (0.04)	74079 (6.53)	74079 (6.66)	74079 (6.66)	74079 (6.43)	74079 (5.44)	74079 (5.21)
R01,F05,C1	92403 (0.04)	92403 (6.13)	92403 (4.95)	92403 (6.37)	92403 (6.13)	92403 (5.43)	92403 (5.36)
R01,F10,C1	115304 (0.09)	115304 (6.28)	115304 (4.76)	115304 (6.40)	115304 (5.32)	115304 (5.5)	115304 (5.35)
R01,F01,C2	84908 (0.46)	84908 (7.62)	84908 (6.39)	85066.67 (8.70)	84908 (6.34)	84908 (7.03)	84908 (7.23)
R01,F05,C2	113036 (1.35)	113036 (9.84)	113036 (7.42)	113036 (9.36)	113036 (7.45)	113036 (7.04)	113036 (7.56)
R01,F10,C2	147599 (2.36)	147599 (8.62)	147599 (7.74)	147796.67 (9.16)	147599 (7.84)	147599 (7.18)	147599 (7.32)

TAB. 5.7 – Computational Results R01 :10,25,10

PROB	OPT	TC	PR	AV.TC	AV.PR	LS-NODE	LS-ARC
R02,F01,C1	232239	232239	232239	232239	232239	232239	232239
	(0.31)	(16.21)	(13.03)	(17.48)	(14.76)	(22.76)	(24.66)
R02,F05,C1	322453	328005	326597	330429.00	327066.33	324852	322453
	(1.12)	(24.67)	(20.09)	(24.89)	(23.56)	(23.87)	(24.32)
R02,F10,C1	419503	439687	419503	440508.00	426999.66	423741	420018
	(1.37)	(25.31)	(20.69)	(25.51)	(26.43)	(25.31)	(24.21)
R02,F01,C2	316437	316437	316437	316437.00	316437	316437	316437
	(0.23)	(25.55)	(21.92)	(25.86)	(22.12)	(25.04)	(25.23)
R02,F05,C2	431250	432234	431734	433979.00	432470	431734	437970
	(0.53)	(29.45)	(23.28)	(29.85)	(26.34)	(25.46)	(25.86)
R02,F10,C2	559578	569032	559578	571400.67	560082.33	563419	563419
	(0.55)	(27.32)	(23.56)	(29.06)	(24.56)	(26.06)	(26.54)

TAB. 5.8 – Computational Results R02 :10,25,25

PROB	OPT	TC	PR	AV.TC	AV.PR	LS-NODE	LS-ARC
R03,F01,C1	484830	484830	484830	484830	485678.33	485406	485406
	(0.67)	(52.46)	(58.16)	(53.26)	(54.23)	(90.39)	(87.57)
R03,F05,C1	703362	716524	704893	718578.33	706213	706787	704947
	(2.01)	(60.65)	(58.17)	(66.84)	(60.23)	(87.24)	(87.67)
R03,F10,C1	944990	985909	962408	992413.67	964399	958256	958256
	(2.34)	(75.81)	(58.25)	(76.22)	(60.12)	(84.29)	(84.87)
R03,F01,C2	704247	705207	705207	705207.00	705207	704247	704247
	(0.47)	(89.35)	(46.86)	(89.88)	(50.23)	(82.79)	(87.72)
R03,F05,C2	932897	942929	939055	943995.00	939372.66	937865	937865
	(1.26)	(89.41)	(67.14)	(90.34)	(70.23)	(79.83)	(83.16)
R03,F10,C2	1188638	1205740	1197950	1224473.33	1198550	1193120	1193120
	(0.98)	(90.14)	(60.16)	(91.32)	(64.12)	(66.29)	(76.89)

TAB. 5.9 – Computational Results R03 :10,25,50

PROB	OPT	TC	PR	AV.TC	AV.PR	LS-NODE	LS-ARC
R04,F01,C1	31730	31730	31730	31730	31730	31730	31730
	(0.02)	(9.51)	(6.84)	(9.59)	(7.4)	(6.69)	(6.67)
R04,F05,C1	48920	48920	48920	48920	48920	48920	48920
	(0.06)	(8.85)	(6.28)	(9.52)	(7.34)	(6.64)	(6.53)
R04,F10,C1	63767	63767	63767	63767	63767	63767	63767
	(0.06)	(9.75)	(7.36)	(10.15)	(8.43)	(6.9)	(6.46)
R04,F01,C2	33740	33740	33740	33740	33740	33740	33740
	(0.57)	(9.85)	(7.4)	(10.19)	(8.76)	(6.39)	(6.39)
R04,F05,C2	53790	53790	53790	53790	53790	53884	53790
	(2.62)	(10.39)	(7.62)	(10.65)	(7.45)	(6.81)	(6.98)
R04,F10,C2	74030	74030	74030	74030	74030	74484	74030
	(1.72)	(10.37)	(8.18)	(10.76)	(8.46)	(6.69)	(6.73)
R04,F01,C8	68291.7	68517	68310	68551.67	68456	68661	68291.7
	(5.14)	(13.27)	(11.45)	(14.53)	(12.53)	(9.64)	(10.04)
R04,F05,C8	113004	113226	113226	113744	113226	113226	113226
	(9.41)	(15.79)	(11.69)	(15.99)	(12.56)	(9.49)	(9.74)
R04,F10,C8	163208	165657	164091	166764.67	164317	164430	163476
	(7.61)	(15.95)	(11.54)	(16.27)	(13.34)	(9.62)	(9.68)

TAB. 5.10 – Computational Results R04 :10,50,10

PROB	OPT	TC	PR	AV.TC	AV.PR	LS-NODE	LS-ARC
R05,F01,C1	123003	123109	123003	123465.67	123199.66	123455	123561
	(0.38)	(23.48)	(17.94)	(23.61)	(20.21)	(18.22)	(18.78)
R05,F05,C1	170060	172517	170467	173327.67	171573	170467	170467
	(1.51)	(24.24)	(18.4)	(24.44)	(18.45)	(20.63)	(19.47)
R05,F10,C1	221486	225950	221486	226440.00	221967.33	**221486**	221486
	(8.44)	(25.57)	(19.78)	(25.70)	(20.4)	(22.56)	(22.07)
R05,F01,C2	131608	131888	131608	132153.33	131662.66	131625	131625
	(1.58)	(26.55)	(20.38)	(26.89)	(20.67)	(19.85)	(20.12)
R05,F05,C2	204157	206497	205764	207021.33	205764	204157	204593
	(15.60)	(28.49)	(21.53)	(28.90)	(23.43)	(21.09)	(21.33)
R05,F10,C2	286524	290574	290574	293937.67	292887.66	288630	287252
	(65.12)	(28.57)	(22.9)	(30.12)	(22.8)	(21.47)	(21.7)
R05,F01,C8	278372	278372	278372	278372	278372	278372	278372
	(1.45)	(43.70)	(47.25)	(52.08)	(49.32)	(43.42)	(47.53)
R05,F05,C8	445810	448542	449133	449319.00	446986.33	452128	451408
	(1.99)	(50.45)	(45.83)	(53.83)	(47.9)	(40.68)	(42.31)
R05,F10,C8	625879	628628	626291	630216.00	628123.66	634657	637466
	(2.20)	(42.50)	(39.65)	(46.58)	(40.45)	(38.31)	(39.28)

TAB. 5.11 – Computational Results R05 :10,50,25

PROB	OPT	TC	PR	AV.TC	AV.PR	LS-NODE	LS-ARC
R06,F01,C1	245936	247829	246365	248527.67	247028.66	247282	247282
	(3.97)	(58.64)	(48.84)	(59.55)	(49.87)	(58.53)	(60.08)
R06,F05,C1	401685	410457	405672	414565.67	407483.66	404317	404317
	(51.12)	(71.69)	(57.08)	(72.96)	(60.12)	(66.59)	(66.6)
R06,F10,C1	559477	587827	572112	592116.33	576620.33	566894	566894
	(205.18)	(69.56)	(61.74)	(73.04)	(59.24)	(73.07)	(69.39)
R06,F01,C2	286682	288200	287510	288861.67	287551	287086	287086
	(25.18)	(74.92)	(62.51)	(79.97)	(62.4)	(73.2)	(70.86)
R06,F05,C2	498266	517614	501560	519212.67	512575.33	500382	500382
	(463.4)	(94.42)	(71.95)	(97.03)	(73.67)	(74.84)	(75.55)
R06,F10,C2	734414	775792	753337	783014.67	763504	745900	752851
	(1855.94)	(99.10)	(74.51)	(100.03)	(71.08)	(77.98)	(75.81)
R06,F01,C8	682921	683481	683614	683525.33	683692.66	683614	683614
	(1.47)	(172.53	(125.96)	(177.05)	(126.87)	(137.31)	(143.75)
R06,F05,C8	1030479	1042510	1038600	1044586.67	1041900	1036980	1036980
	(1.45)	(176.18)	(130.92)	(190.94)	(133.98)	(114.19)	(117.63)
R06,F10,C8	423316	431329	433690	442105.33	440456.33	440374	440459
	(46.56)	(54.54)	(45.34)	(56.35)	(45.45)	(62.05)	(59.4)

TAB. 5.12 – Computational Results R06 :10,50,50

PROB	OPT	TC	PR	AV.TC	AV.PR	LS-NODE	LS-ARC
R07,F01,C1	32807	32807	32807	32807	32807	32807	32807
	(0.08)	(15.65)	(14.00)	(14.99)	(13.6)	(8.92)	(8.72)
R07,F05,C1	47252	47252	47252	47252	47252	47252	47252
	(0.13)	(12.37)	(8.8)	(12.61)	(9.21)	(8.52)	(8.58)
R07,F10,C1	62962	62962.0	62962	62962	62987.66	62962	62962
	(0.62)	(15.17)	(9.62)	(15.42)	(10.34)	(8.78)	(8.96)
R07,F01,C2	37432	37432.0	37432	37432	37432	37432	37432
	(0.66)	(13.76)	(10.06)	(13.89)	(10.45)	(9.99)	(9.79)
R07,F05,C2	56475	56841.0	56475	56841	56552.33	56475	56475
	(4.39)	(15.05)	(10.74)	(15.18)	(11.43)	(9.02)	(9.35)
R07,F10,C2	77249	77863.0	77249	78842.67	78489.66	78048	77863
	(19.01)	(14.20)	(10.61)	(15.32)	(10.45)	(9.08)	(9.65)
R07,F01,C8	59947	59947.0	59947	59947	60061.66	60184	60184
	(11.05)	(17.99)	(15.03)	(19.62)	(18.62)	(14.13)	(13.63)
R07,F05,C8	99194	102203.	99502	102747.33	99757.33	101860	103906
	(21.75)	(20.63)	(14.52)	(21.26)	(14.23)	(12.99)	(12.68)
R07,F10,C8	141692	144329.	141788	144079	143898.33	147147	146296
	(54.39)	(19.16)	(14.84)	(20.62)	(15.23)	(12.77)	(12.87)

TAB. 5.13 – Computational Results R07 :10,75,10

PROB	OPT	TC	PR	AV.TC	AV.PR	LS-NODE	LS-ARC
R08,F01,C1	102531	102556	102531	102880.00	102547.66	102531	102531
	(1.33)	(31.60)	(26.15)	(32.02)	(25.14)	(27.23)	(27.98)
R08,F05,C1	143894	145434	143894	145818.67	143894	143894	143894
	(0.54)	(31.53)	(24.62)	(31.93)	(24.45)	(27.5)	(27.5)
R08,F10,C1	182793	182793	182793	182969.00	184068.33	183321	186022
	(1.35)	(31.83)	(26.29)	(31.91)	(25.34)	(31.47)	(28.39)
R08,F01,C2	109325	110193	109325	110308.33	109358.33	109325	109325
	(5.45)	(34.28)	(25.45)	(34.64)	(26.45)	(25.55)	(25.87)
R08,F05,C2	157047	160848	157901	161447.67	158039	157720	157720
	(34.23)	(35.76)	(27.7)	(36.36)	(28.8)	(27.09)	(26.85)
R08,F10,C2	207540	214793	208325	214993.00	209766	209433	209433
	(34.23)	(34.52)	(27.95)	(35.79)	(27.7)	(28.49)	(29.45)
R08,F01,C8	154160	156738	154327	157065.33	155292.33	155166	155604
	(75.38)	(54.46)	(38.12)	(55.01)	(39.13)	(35.33)	(35.1)
R08,F05,C8	274866.5	282525	278443	287224.33	280858.33	281110	277028
	(404.49)	(53.88)	(42.52)	(55.15)	(41.15)	(36.06)	(35.18)
R08,F10,C8	415793	436669	421292	444136.00	427881	423783	428095
	(776.56)	(54.40)	(41.33)	(56.67)	(43.8)	(36.59)	(37.31)

TAB. 5.14 – Computational Results R08 :10,75,25

PROB	OPT	TC	PR	AV.TC	AV.PR	LS-NODE	LS-ARC
R09,F01,C1	171512	171902	171572	172382.67	171655.33	171512	171512
	(2.33)	(71.94)	(55.89)	(74.39)	(56.45)	(70.94)	(75.44)
R09,F05,C1	296712	305497	302448	308971.67	303722	305856	306865
	(32.65)	(66.57)	(66.94)	(76.45)	(66.56)	(74.05)	(73.42)
R09,F10,C1	424266	435762	431584	440892.33	436532.66	432442	438435
	(218.77)	(73.17)	(64.81)	(80.40)	(65.56)	(80.79)	(82.01)
R09,F01,C2	192736	193734	192736	194155.67	193203.66	195641	194426
	(10.2)	(80.08)	(67.11)	(83.27)	(69.13)	(76.48)	(76.91)
R09,F05,C2	357318	368592	366418	373219.67	371407.66	371897	367341
	(143.32)	(90.21)	(76.96)	(99.48)	(79.45)	(88.12)	(82.31)
R09,F10,C2	522187	555768	533927	560263.67	543955	538298	538298
	(1227.52)	(86.08)	(80.26)	(93.20)	(83.98)	(88.11)	(84.39)
R09,F01,C8	345057	348732	346773	349684.33	348735.33	348250	348118
	(27.83)	(151.35)	(124.73)	(153.15)	(125.87)	(114.83)	(109.77)
R09,F05,C8	646579	668296	657541	669379.00	665286.33	661744	661744
	(89.5)	(156.57)	(117.78)	(161.14)	(127.54)	(108.48)	(110.9)
R09,F10,C8	951136	987258.	979672	992832.00	980308.66	993152	997409
	(310.59)	(155.66)	(123.66)	(159.08)	(133.98)	(108.18)	(115.18)

TAB. 5.15 – Computational Results R09 :10,75,50

PROB	OPT	TC	PR	AV.TC	AV.PR	LS-NODE	LS-ARC
R10,F01,C1	200087 (10.72)	200484 (104.95)	200147 (75.22)	200542.33 (106.74)	200340.33 (76.34)	200087 (77.93)	200087 (79.5)
R10,F05,C1	346813.5 (788.2)	350407 (110.20)	347459 (85.65)	352366.00 (116.93)	347835.66 (88.56)	349386 (78.32)	348608 (85.37)
R10,F10,C1	488015 (563.29)	502724 (127.87)	502724 (84.8)	505694.00 (128.99)	504601 (84.56)	511951 (87.6)	506559 (85.73)
R10,F01,C2	229196 (184.31)	231302 (128.79)	229196 (90.6)	232220.67 (134.77)	229363 (97.56)	229939 (83.3)	229735 (84.73)
R10,F05,C2	411664 (5123.25)	430254 (149.47)	425230 (102.63)	433229.00 (149.82)	426363.66 (110.34)	428633 (87.47)	421339 (95.91)
R10,F10,C2	609104 (27260.53)	646107 (158.59)	628353 (109.82)	664652.33 (159.23)	633081.33 (109.54)	664931 (91.19)	654135 (91.77)
R10,F01,C8	486895 (391.33)	490034 (390.26)	487235 (330.31)	491176.67 (398.56)	487923.33 (343.89)	490315 (321.11)	489337 (335.3)
R10,F05,C8	951056 (555.52)	977346 (286.08)	963914 (274.29)	980466.67 (315.74)	967753.33 (284.76)	975315 (265.41)	977044 (245.84)
R10,F10,C8	1421740 (608.38)	1469020 (228.92)	1439320 (248.12)	1491893.33 (243.37)	1448780 (245.78)	1488300 (218.16)	1465820 (237.72)

TAB. 5.16 – Computational Results R10 :20,100,40

PROB	OPT	TC	PR	AV.TC	AV.PR	LS-NODE	LS-ARC
R11,F01,C1	714431 (392.46)	726585 (441.29)	720236 (355.76)	730686.33 (447.77)	724441 (358.43)	724540 (421.58)	724934 (449.75)
R11,F05,C1	1265985 (t)	1323620 (489.85)	1296050 (510.67)	1327736.67 (601.70)	1304816.66 (528.12)	1301970 (551.6)	1301970 (562.08)
R11,F10,C1	1846295 (t)	1933560 (706.75)	1940180 (576.94)	1969320 (723.43)	1951226.66 (587.76)	1935700 (586.15)	1935700 (597.54)
R11,F01,C2	870451 (2960.13)	879445 (819.49)	875908 (550.56)	883224.67 (830.33)	877552 (555.43)	875799 (670.11)	875799 (690.15)
R11,F05,C2	1623640 (21739.5)	1715040 (1137.94)	1674340 (889.13)	1721046.67 (1153.43)	1687013.33 (890.33)	1697950 (841.01)	1674880 (825.76)
R11,F10,C2	2414060 (t)	2593590 (1274.95)	2546730 (990.76)	2600196.67 (1332.79)	2606896.66 (1010.45)	2453890 (768.67)	2516440 (858.24)
R11,F01,C8	2294912 (31.51)	2294912 (880.22)	2294912 (1181.57)	2295296.67 (1088.83)	2294912 (1089.56)	2296070 (1203.88)	2296070 (1300.86)
R11,F05,C8	3507100 (79.79)	3540730 (1174.59)	3529530 (1079.67)	3550923.33 (1235.99)	3539613.33 (1102.99)	3573960 (964.81)	3588220 (1000.82)
R11,F10,C8	4579353 (48.51)	4616190 (827.84)	4592820 (777.86)	4645180 (849.09)	4653570 (812.45)	4673450 (784.87)	4693410 (764.87)

TAB. 5.17 – Computational Results R11 :20,100,100

PROB	OPT	TC	PR	AV.TC	AV.PR	LS-NODE	LS-ARC
R12,F01,C1	1639443 (4436.74)	1700170 (2199.46)	1668370 (2434.57)	1706230 (2720.30)	1677986.66 (2453.3)	1656170 (3547.48)	1661140 (3718.41)
R12,F05,C1	3403074.2 (t)	3741660 (3188.77)	3717800 (2975.35)	3767176.67 (3477.48)	3728346.66 (3177.3)	3690090 (3837.31)	3690090 (4467.24)
R12,F10,C1	5276171 (t)	6055670 (4096.82)	6037960 (3245.49)	6100926.67 (4331.77)	6073666.66 (3476.4)	5558110 (4301.1)	5774190 (4824.2)
R12,F01,C2	2303557 (1739.79)	2331290 (6504.21)	2326930 (5539.91)	2332063.33 (7467.51)	2325936.66 (6432.43)	2330600 (9432.55)	2330090 (7715.68)
R12,F05,C2	4669799 (3145.59)	4942920 (7364.80)	4954250 (5709.11)	4982423.33 (7574.81)	4978036.66 (5634.98)	4886780 (10900.8)	4886780 (8780.59)
R12,F10,C2	7100019 (1683.84)	7562660 (6060.43)	7638050 (4735.91)	7653836.67 (7704.07)	7660990 (5432.98)	7306910 (8918.51)	7306910 (9065.54)
R12,F01,C8	7635270 (78.13)	7638050 (5336.99)	7638050 (4735.91)	7638050 (5787.92)	7638050 (4654.33)	7638490 (5731.17)	7638490 (5585.26)
R12,F05,C8	10067742 (61.55)	10106800 (3176.64)	10085000 (3315.88)	10116466.67 (3347.35)	10104800 (3412.5)	10140400 (3309.39)	10122300 (3309.39)
R12,F10,C8	11967768 (50.29)	12064300 (2459.84)	12042800 (2276.53)	12083133.33 (2503.33)	12063866.67 (2341.4)	12082800 (2454.33)	12072500 (2583.71)

TAB. 5.18 – Computational Results R12 :20,100,200

PROB	OPT	TC	PR	AV.TC	AV.PR	LS-NODE	LS-ARC
R13,F01,C1	142947 (6.41)	143778 (201.55)	142947 (130.35)	144437.33 (202.37)	143139.66 (138.4)	145946 (143.34)	145926 (131.94)
R13,F05,C1	263800 (547.98)	272019 (203.12)	273853 (141.32)	275961.33 (205.89)	273912 (145.32)	274807 (151.06)	278363 (142.5)
R13,F10,C1	365836 (116.62)	375628 (214.58)	378922 (145.79)	379687.67 (221.23)	387201.33 (147.34)	405741 (143.04)	377341 (136.96)
R13,F01,C2	150977 (90.14)	151875 (208.55)	150977 (143.09)	153080 (209.60)	152617.66 (141.23)	153964 (128.61)	153956 (131.04)
R13,F05,C2	282682 (3551.86)	290150 (225.07)	289706 (142.83)	293241.67 (231.63)	294079.33 (144.76)	292676 (142.42)	291331 (137.51)
R13,F10,C2	406790 (29695.59)	430246 (246.71)	435724 (159.9)	435988.33 (252.66)	439691.33 (164.34)	444476 (167.32)	446796 (144.58)
R13,F01,C8	208088 (t)	213650 (319.67)	210898 (183.14)	214628.67 (326.24)	212134 (197.23)	211520 (176.28)	211864 (170.67)
R13,F05,C8	446997 (t)	485018 (379.88)	469035 (182.03)	491599.67 (383.64)	481539.66 (189.45)	481498 (162.63)	480409 (177.8)
R13,F10,C8	698280 (t)	787477 (391.99)	761505 (232.08)	801558.67 (403.09)	773893 (245.76)	804859 (192.32)	789122 (187.67)

TAB. 5.19 – Computational Results R13 :20,200,40

PROB	OPT	TC	PR	AV.TC	AV.PR	LS-NODE	LS-ARC
R14,F01,C1	403414 (495.72)	416932 (561.79)	410430 (413.86)	418294.67 (567.40)	411263.66 (412.56)	419549 (497.14)	419448 (469.5)
R14,F05,C1	750091 (t)	775023 (598.24)	778492 (503.86)	792663.67 (644.45)	787719.66 (576.32)	797655 (639.8)	797655 (645.07)
R14,F10,C1	1078595 (t)	1133420 (723.41)	1127980 (555.53)	1160773.33 (737.84)	1133240 (604.12)	1117240 (605.9)	11890100 (640.54)
R14,F01,C2	437607 (1327.37)	448725 (633.53)	444170 (463.33)	451845.33 (637.85)	445088.33 (475.23)	447920 (567.3)	447779 (609.85)
R14,F05,C2	852330 (t)	890914 (731.71)	883241 (580.59)	897678.00 (777.93)	894028 (590.43)	930149 (776.9)	901734 (756.63)
R14,F10,C2	1230749.7 (t)	1309970 (818.25)	1335170 (608.59)	1332676.67 (833.51)	1340806.66 (673.12)	1328700 (747.16)	13043900 (808.87)
R14,F01,C8	670064 (t)	698381 (1794.67)	689762 (1294.26)	701782.33 (2000.69)	692596.33 (1576.34)	704695 (1187.04)	702252 (1195.67)
R14,F05,C8	1643376 (t)	1782720 (2516.48)	1740600 (1810.25)	1791180.00 (2717.66)	1771183.33 (1965.45)	1812420 (1407.45)	1786510 (1427.45)
R14,F10,C8	2630332 (t)	2901140 (3223.01	2939990 (2555.49)	2911496.67 (3335.62)	3008416.66 (2875.45)	3189060 (1738.36)	3139640 (1709.39)

TAB. 5.20 – Computational Results R14 :20,200,100

PROB	OPT	TC	PR	AV.TC	AV.PR	LS-NODE	LS-ARC
R15,F01,C1	1000787 (5384.72)	1044180 (1698.44)	1027930 (1361.73)	1053690 (1808.50)	1034716.66 (1587.43)	1016090 (2190.69)	1016090 (2178.73)
R15,F05,C1	1979413 (t)	2164870 (2323.56)	2135640 (1871.08)	2183736.67 (2475.63)	2183433.33 (1896.43)	2089670 (2527.68)	2079670 (2544.43)
R15,F10,C1	2949264 (t)	3156520 (2546.50)	3131640 (2106.42)	3178423.33 (2572.87)	3190583.33 (2474.12)	3142150 (2839.41)	3082600 (2702.37)
R15,F01,C2	1148604 (t)	1211560 (2593.89)	1184780 (2049.64)	1217346.67 (2904.63)	1192780 (2341.23)	1179490 (4361.04)	1179490 (4256.61)
R15,F05,C2	2488753 (t)	2736570 (4999.33)	2732310 (5114.44)	2747943.33 (5778.15)	2771770 (5045.12)	2674960 (6313.28)	2779670 (5374.21)
R15,F10,C2	3972667 (t)	4393650 (6171.74)	4264090 (4594.46)	4395146.67 (6846.45)	4304980 (4653.12)	4279020 (6175.27)	4225850 (5537.29)
R15,F01,C8	2301326.6 (t)	2355980 (18162.70)	2350730 (14515.9)	2360816.67 (18763.03)	2351980 (14875.9)	2351610 (16402.9)	2349960 (15883.2)
R15,F05,C8	5573412.8 (16107.92)	5941080 (20972.30)	5830430 (23169.4)	5944116.67 (24038.43)	5858273.33 (22453.34)	5885100 (20359.7)	5898640 (19392.1)
R15,F10,C8	8696932 (18930.96)	9140060 (14527.20)	9123020 (19548.3)	9220093.33 (15536.40)	9131563.33 (18765.76)	9193230 (17950.1)	9203600 (19460.2)

TAB. 5.21 – Computational Results R15 :20,200,200

PROB	OPT	TC	PR	AV.TC	AV.PR	LS-NODE	LS-ARC
R16,F01,C1	136161	136851	136161	137553.00	136763.66	137776	137956
	(1.05)	(322.20)	(194.43)	(327.58)	(190.34)	(174.4)	(191.51)
R16,F05,C1	239500	245003	244070	248182.00	248259.33	262416	261514
	(1641.30)	(309.91)	(200.35)	(311.99)	(234.56)	(241.36)	(210.41)
R16,F10,C1	325671	346724	330138	348585.67	342357.33	347035	358362
	(4345.92)	(306.85)	(212.21)	(312.22)	(240.34)	(206.86)	(196.83)
R16,F01,C2	138532	140433	139131	141125.33	139996.66	140494	140244
	(27.61)	(327.39)	(184.9)	(328.64)	(203.54)	(197.03)	(184.55)
R16,F05,C2	241801	250332	248717	253838.67	254805.66	256552	260552
	(851.99)	(325.14)	(204.65)	(329.91)	(198.45)	(216.56)	(195.97)
R16,F10,C2	337762	352450	343165	357835.33	357759	375041	365785
	(8173.47)	(305.98)	(206.89)	(315.81)	(250.45)	(252.73)	(254.73)
R16,F01,C8	169502	173164	171651	174503.67	172527	173620	172030
	(t)	(382.96)	(209.24)	(387.08)	(278.45)	(194.56)	(204.96)
R16,F05,C8	352976	374372	371129	376533.67	372825	378228	393434
	(t)	(429.56)	(244.07)	(434.95)	(290.45)	(241.42)	(222.55)
R16,F10,C8	541626	587242	564151	590970.00	575334	631452	621697
	(t)	(450.30)	(229.92)	(456.95)	(249.45)	(220.93)	(258.93)

TAB. 5.22 – Computational Results R16 :20,300,40

PROB	OPT	TC	PR	AV.TC	AV.PR	LS-NODE	LS-ARC
R17,F01,C1	354138	367923	364325	370314.67	365378.66	368730	368730
	(332.10)	(790.06)	(536.69)	(810.44)	(654.34)	(797.94)	(808.7)
R17,F05,C1	651025	693482	669935	696950.33	687806.33	700063	676886
	(t)	(822.09)	(710.92)	(866.93)	(806.45)	(761.51)	(798.78)
R17,F10,C1	917956	990967	973486	996346.00	980134.66	973449	991618
	(t)	(871.10)	(845.3)	(948.55)	(765.52)	(844.49)	(794.68)
R17,F01,C2	370590	382868	378905	385026.33	380997	388354	386287
	(2258.82)	(866.99)	(615.05)	(899.28)	(654.9)	(827.04)	(824.57)
R17,F05,C2	708698	761476	743173	768713.67	748048.66	773392	759366
	(t)	(988.79)	(713.2)	(1026.86)	(750.54)	(948.2)	(888.64)
R17,F10,C2	1029242	1104380	1084280	1116650.00	1098420	1075040	1095890
	(t)	(1092.96)	(832.08)	(1134.30)	(965.87)	(837.05)	(824.25)
R17,F01,C8	505310	530313	521468	532371.00	523786	533966	528988
	(t)	(1622.41)	(1026.25)	(1691.09)	(1087.65)	(881.53)	(971.76)
R17,F05,C8	1112076	1249440	1191580	1265813.33	1209456.6	1281960	1255980
	(t)	(2313.01)	(1486.98)	(2774.21)	(1643.98)	(1201.77)	(1178.32)
R17,F10,C8	1847334	2017720	1985220	2035663.33	2014533.3	2117460	2140180
	(t)	(3048.11)	(2300.72)	(3165.96)	(2654.43)	(1384.99)	(1408.5)

TAB. 5.23 – Computational Results R17 :20,300,100

PROB	OPT	TC	PR	AV.TC	AV.PR	LS-NODE	LS-ARC
R18,F01,C1	828559	866838	858043	877666.67	861656.66	861193	860604
	(t)	(2167.16)	(1926.27)	(2235.21)	(2076.54)	(2432.72)	(2470.87)
R18,F05,C1	1542197	1671810	1649730	1726433.33	1712110	1630260	1666160
	(t)	(2609.61)	(2279.48)	(2671.03)	(2465.3)	(2571.32)	(2653.25)
R18,F10,C1	2292684	2383760	2418190	2457733.33	2451633.33	2265930	2334580
	(t)	(2659.54)	(2167.09)	(2779.20)	(2298.9)	(2679.52)	(2697.9)
R18,F01,C2	921762	980992	964367	986701.67	966850.66	945374	951226
	(t)	(2717.04)	(2209.13)	(2810.17)	(2564.34)	(3130.84)	(3090.22)
R18,F05,C2	1866215	2033500	1980170	2042536.67	2018590	2037290	2037290
	(t)	(3865.35)	(2810.98)	(4168.55)	(3032.4)	(4054.3)	(4676.41)
R18,F10,C2	2895982	2981980	2928930	3011723.33	2972860	2969700	2964360
	(t)	(3828.76)	(3575.27)	(3939.71)	(3487.5)	(4450.03)	(4003.13)
R18,F01,C8	1485690.8	1611800	1568050	1622236.67	1576570	1588930	1599030
	(t)	(15115.90)	(8151.71)	(19064.10)	(10321.23)	(6751.37)	(7151.65)
R18,F05,C8	4010736.7	4379870	4553560	4515690.00	4690270	4216690	4303900
	(t)	(33140.40)	(15068.2)	(34565.93)	(18654.4)	(13743.3)	(15232.1)
R18,F10,C8	6663506.4	7562480	7639160	7762600.00	7998752.33	6907100	6842610
	(t)	(31716.60)	(26296.6)	(32354.43)	(28642.6)	(11844.5)	(10815.5)

TAB. 5.24 – Computational Results R18 :20,300,200

Chapitre 6

Conclusion

Cette thèse est consacrée à l'étude du problème de synthèse de réseau multiproduits avec capacités. Ce problème est important non seulement en raison de son application en planification de réseaux de transport et de télécommunication mais également, parce qu'il pose des défis considérables. De ce fait, nous nous sommes intéressés dans cette thèse au développement de méthodes de recherche avec tabous pour fournir des solutions de bonne qualité pour ce problème.

Dans ce qui suit, nous présentons un résumé des contributions principales de cette thèse et nous dégageons des avenues de recherche ouvertes à la suite de ce travail.

6.1 Contributions

Dans le chapitre 2, nous avons présenté la méthode de recherche avec tabous et certaines de ses stratégies de recherche. Nous avons également présenté une synthèse des réalisations les plus pertinentes pour le problème de synthèse de réseau multiproduits avec capacités et des méthodologies générales adaptées pour sa résolution.

Dans le chapitre 3, nous avons proposé une nouvelle structure de voisinage pour les méta-heuristiques dédiées au problème de synthèse de réseau multiproduits avec capacités. Cette structure de voisinage est basée sur des cycles qui seront détectés et évalués dans des graphes résiduels. L'idée fondamentale est de définir des mouvements qui affectent plusieurs produits en même temps et qui ouvrent et ferment plusieurs arcs à la fois. Le mouvement consiste donc dans la déviation du flot total d'un chemin à un autre. Ces deux chemins relient deux noeuds quelconques du réseau (ne sont pas restreints à relier l'origine et la destination d'un produit) et forment un cycle. La

déviation du flot autour de ce cycle résultera dans l'ouverture et la fermeture de certains arcs du cycle. Nous avons développé une méthode simple de recherche avec tabous pour tester cette structure de voisinage. Les résultats obtenus ont indiqué clairement la supériorité de cette approche par rapport à la meilleure heuristique proposée dans la littérature.

Dans le chapitre 4, nous avons proposé une procédure de "path relinking" pour notre problème. Des solutions élites sont construites en exécutant un simple algorithme de recherche avec tabous qui utilise des structures de voisins basées sur des cycles. Une version restreinte de la même structure de voisinage est utilisée pour explorer le chemin entre deux solutions élites candidates. Nous avons développé et comparé plusieurs variantes de la méthode. Les résultats expérimentaux ont indiqué que cette méthode constitue la meilleure heuristique actuellement disponible pour cette classe de problèmes.

Le chapitre 5 a porté sur l'étude du mécanisme d'apprentissage. Notre motivation principale était d'explorer la portée de ce concept qui nous semblait bénéfique pour l'élaboration de meilleures solutions. De ce fait, nous avons développé et examiné des procédures d'intensification et de diversification utilisant des structures de mémoires adaptatives pour notre problème. Finalement, nous avons évalué les avantages de son utilisation.

6.2 Nouvelles avenues de recherche

Plusieurs travaux de recherche s'ouvrent suite à ce travail. Parmi ces nouveaux développements, mentionnons :

- L'extension des travaux pour le problème de synthèse de réseau de service, où les variables entières ne sont plus restreintes à des valeurs 0 ou 1. En effet, une telle extension présente de nouveaux défis. Par exemple, dans le cas de réseaux de service, il faudrait sans doute considérer les fréquences sur les arcs. Par ailleurs, il faut généraliser la structure du voisinage, ainsi que la construction du graphe résiduel présentés au chapitre 3 pour tenir compte de ce fait. A cette fin, il ne faut plus mettre en question la fermeture ou l'ouverture de l'arc mais plutôt de se questionner sur l'ajout ou l'élimination d'un service sur cet arc. Ce nouveau

concept englobe la structure de voisinage présenté au chapitre 3. En effet, lorsque le nombre de services est égal à 1, ajouter ou éliminer un service revient à ajouter ou éliminer un arc. De plus, la réalisation de cette extension serait faisable en modifiant la structure du graphe résiduel en introduisant la notion de service sur les arcs.

- Le parallélisme s'avère un outil important pour traiter les problèmes de grande taille. Il serait intéressant d'étudier l'impact du paralléllisme sur les travaux développés dans cette thèse.
- La mise au point d'algorithmes de séparation et évaluation progressive ("branch-and-bound"), l'ajout des inégalités valides et leur intégration avec les travaux de cette thèse serait un pas remarquable vers l'élaboration des solutions optimales "exactes" pour le problème de synthèse de réseau multiproduits avec capacités.
- Les problèmes de synthèse dynamique constituent une classe de problèmes extrêmement difficiles et peu étudiés dans la littérature. Dans ce cas, il faut tenir compte des aspects stochastiques du problème comme l'évolution des offres et des demandes (nouvelles demandes qui surgissent, clients qui annulent leur demande, ...) sur un horizon de planification assez long. Il serait judicieux en premier lieu d'étendre et de généraliser les travaux de cette thèse pour traiter ces problèmes.

Bibliographie

Ahuja, R.K., Magnanti, T.L., and Orlin, J.B. (1993). *Network Flows – Theory, Algorithms, and Applications.* Prentice-Hall, Englewood Cliffs, NJ.

Balakrishnan, A., Magnanti, T.L., and Mirchandani, P. (1997). Network Design. In Dell'Amico, M., Maffioli, F., and Martello, S., editors, *Annotated Bibliographies in Combinatorial Optimization*, pages 311–334. John Wiley & Sons, New York, NY.

Balakrishnan, A., Magnanti, T.L., and Wong, R.T. (1989). A Dual–Ascent Procedure for Large–Scale Uncapacitated Network Design. *Operations Research*, 37(5) :716–740.

Barahona F. (1996). Network Design Using Cut Inequalities. *SIAM Journal of Optimization*, 6 :823–837.

Bazlamaçci C. and Hindi K. S. (1996). Enhanced Adjacent Extreme Point Search and Tabu Search for the Minimum, Concave-Cost Uncapacitated Transshipment Problem. *Journal of the Operational Research Society*, 47 :1150–1165.

Berger, D., Gendron, B., Potvin, J.Y., Raghavan, S., and Soriano, P. (2000). Tabu Search for a Network Loading Problem with Multiple Facilities. *Journal of Heuristics*, 6(2) :253–267.

Bienstock D. and Günlük O. (1996). Capacitated Network Design-Polyedral Structure and Computation. *INFORMS Journal on Computing*, 8 :243–259.

Chopra S., Gilboa I., and Sastry S.T. (1998). Source Sink Flows with Capacity Installation in Batches. *Discrete Applied Mathematics*, 85 :165–192.

Chouman, M., Crainic, T.G., and Gendron, B. (2003a). A Cutting-Plane Algorithm Based on Cutset Inequalities for Multicommodity Capacitated Fixed Charge Net-

work Design . Technical report, Centre de recherche sur les transports, Université de Montréal, Montréal, QC, Canada.

Chouman, M., Crainic, T.G., and Gendron, B. (2003b). Revue des inégalités valides pertinentes aux problèmes de conception de réseaux. *INFOR*, 41 :5–33.

Cornuéjols, G., Sridharan, R., and Thizy, J.M. (1991). A Comparison of Heuristics and Relaxations for the Capacitated Plant Location Problem. *European Journal of Operational Research*, 50 :280–297.

Crainic, T.G., Dejax, P.J., and Delorme, L. (1989). Models for Multimode Multicommodity Location Problems with Interdepot Balancing Requirements. *Annals of Operations Research*, 18 :279–302.

Crainic, T.G. and Delorme, L. (1993). Dual-Ascent Procedures for Multicommodity Location-Allocation Problems with Balancing Requirements. *Transportation Science*, 27(2) :90–101.

Crainic, T.G., Ferland, J.-A., and Rousseau, J.-M. (1984). A Tactical Planning Model for Rail Freight Transportation. *Transportation Science*, 18(2) :165–184.

Crainic, T.G., Frangioni, A., and Gendron, B. (2001). Bundle-Based Relaxation Methods for Multicommodity Capacitated Network Design. *Discrete Applied Mathematics*, 112 :73–99.

Crainic, T.G., Gendreau, M., and Farvolden, J.M. (2000). A Simplex-Based Tabu Search Method for Capacitated Network Design. *INFORMS Journal on Computing*, 12(3) :223–236.

Crainic, T.G., Gendreau, M., Soriano, P., and Toulouse, M. (1993). A Tabu Search Procedure for Multicommodity Location/Allocation with Balancing Requirements. *Annals of Operations Research*, 41 :359–383.

Crainic, T.G, Gendron, B, and Hernu, G. (2002). A slope Scaling/Lagrangean Perturbation Heuristic with Long-Term Memory for Multicommodity Capacitated Fixed-Charge Network Design. Publication, Centre de recherche sur les transports, Université de Montréal, Montréal, QC, Canada.

Crainic, T.G. and Laporte, G. (1997). Planning Models for Freight Transportation. *European Journal of Operational Research*, 97(3) :409–438.

Crainic, T.G. and Rousseau, J.-M. (1986). Multicommodity, Multimode Freight Transportation : A General Modeling and Algorithmic Framework for the Service Network Design Problem. *Transportation Research B : Methodology*, 20B :225–242.

Epstein, R. (1998). *Linear Programming and Capacitated Network Loading.* PhD thesis, Sloan School of Management, Massachusetts Institute of Technology.

Farvolden, J.M. and Powell, W.B. (1994). Subgradient Methods for the Service Network Design Problem. *Transportation Science*, 28(3) :256–272.

Gallo G. and Sodini C. (1979). Adjacent Extreme Flows and Application to Min Concave Cost Flow Problem. *Networks*, 9 :95–221.

Garey, M. and Johnson, D. (1979). *Computers and Intractability : A Guide to the Theory of NP-Completeness.* Freeman, San Francisco, CA.

Gendreau, M., Hertz, A., and Laporte, G. (1994). A Tabu Search Heuristic for the Vehicle Routing Problem. *Management Science*, 40 :1276–1290.

Gendron, B., Crainic, T.G., and Frangioni, A. (1998). Multicommodity Capacitated Network Design. In Sansó, B. and Soriano, P., editors, *Telecommunications Network Planning*, pages 1–19. Kluwer Academic Publishers, Norwell, MA.

Gendron, B. and Crainic, T.G. (1994). Relaxations for Multicommodity Network Design Problems. Publication CRT-965, Centre de recherche sur les transports, Université de Montréal, Montréal, QC, Canada.

Gendron B. and Crainic T.G. (1995). A Branch-and-Bound Algorithm for Depot Location and Container Fleet Management. *Location Science*, 3 :39–53.

Gendron, B. and Crainic, T.G. (1996). Bounding Procedures for Multicommodity Capacitated Network Design Problems. Publication CRT-96-06, Centre de recherche sur les transports, Université de Montréal, Montréal, QC, Canada.

Gendron, B. and Crainic, T.G. (1997). A Parallel Branch-and-Bound Algorithm for Multicommodity Location with Balancing Requirements. *Computers & Operations Research*, 24(9) :829–847.

Gendron B., Potvin J.Y., and Soriano P. (1999). Tabu Search with Exact Neighbor Evaluation for Multicommodity Location with Balancing Requirements. *INFOR*, 37 :255–270.

Gendron B., Potvin J.Y., and Soriano P. (2003). A Tabu Search with Slope Scaling for the Multicommodity Capacitated Location Problem with Balancing Requirements. *Annals of Operations Research*, 122 :193–217.

Ghamlouche, I., Crainic, T.G., and Gendreau, M. (2003). Cycle-based Neighbourhoods for Fixed-Charge Capacitated Multicommodity Network Design. *Operations Research*, 51(4) :655–667.

Ghamlouche, I., Crainic, T.G., and Gendreau, M. (2004). Path Relinking, Cycle-based Neighbourhoods and Capacitated Multicommodity Network Design. *Annals of Operations Research*. to appear.

Glover, F., Laguna, M., and Marti, R. (2000). Fundamentals of scatter search and path relinking. *Control and Cybernetics*, 39(3) :653–684.

Glover, F. and Laguna, M. (1997). *Tabu Search*. Kluwer Academic Publishers, Norwell, MA.

Glover, F., Taillard, É.D., and de Werra, D. (1993). A user's guide to tabu search. *Annals of Operations Research*, 41 :3–28.

Glover, F. (1986). Future Paths for Integer Programming and Links to Artificial Intelligence. *Computers & Operations Research*, 1(3) :533–549.

Glover, F. (1989). Tabu Search – Part I. *ORSA Journal on Computing*, 1(3) :190–206.

Glover, F. (1990). Tabu Search – Part II. *ORSA Journal on Computing*, 2(1) :4–32.

Glover, F. (1994). Genetic Algorithms and Scatter Search : Unsuspected Potentials. *Statistics and Computing*, 4 :131–140.

Glover, F. (1996). Tabu Search and Adaptive Memory Programming – Advances, Applications and Challenges. In Barr, R., Helgason, R., and Kennington, J., editors, *Interfaces in Computer Science and Operations Research*, pages 1–75. Kluwer Academic Publishers, Norwell, MA.

Glover, F. (1997). A Template for Scatter Search and Path Relinking. In Hao, J., Lutton, E., Ronald, E., Schoenauer, M., and Snyers, D., editors, *Artificial Evolution*, volume 1363 of *Lecture Notes in Computer Science*, pages 13–54. Springer Verlag, Berlin.

Guisewite, G.M. and Pardalos, P.M. (1990). Minimum concave cost network flow problems : applications, complexity and algorithms. *Annals of Operations Research*, 25 :75–100.

Hellstrand J., Larsson T., and Migdalas A. (1992). A Characterization of the Uncapacitated Network Design Polytope. *Operations Research Letters*, 12 :159–163.

Holmberg, K. and Hellstrand, J. (1998). Solving the Uncapacitated Network Design Problem by a Lagrangian Heuristic and Branch-and-Bound. *Operations Research*, 46(2) :247–259.

Holmberg, K. and Yuan, D. (2000). A Lagrangean Heuristic Based Branch-and-Bound Approach for the Capacitated Network Design Problem. *Operations Research*, 48(3) :461–481.

ILOG (1999). *ILOG CPLEX 6.5.* ILOG, Mountain View, CA. U.S.A.

ILOG (2002). *ILOG CPLEX 7.5.* ILOG, Mountain View, CA. U.S.A.

Jarvis, J.J. and Mejia de Martinez, O. (1977). A Sensitivity Analysis of Multicommodity Network Flows. *Transportation Science*, 11(4) :299–306.

Kelly, J.P., Laguna, M., and Glover, F. (1994). A Study of Diversification Strategies for the Quadratic Assignment Problem. *Computers & Operations Research*, 21(8) :885–893.

Koskosidis, Y.A., Powell, W.B., and Solomon, M.M. (1992). An Optimization–based Heuristic for Vehicle Routing and Scheduling with Soft Time Windows Constraints. *Transportation Science*, 26 :69–85.

Krarup J. and Pruzan P.M. (1983). The Simple Plant Location Problem : Survey and Synthesis. *European Journal of Operational Research*, 12 :36–81.

Laguna, M. and Armentano, V.A. (2001). Lessons from Applying and Experimenting with Scatter Search. In Rego, C. and Alidaee, B., editors, *Adaptive Memory*

and Evolution : Tabu Search and Scatter Search. Kluwer Academic Publishers, Norwell, MA. to appear.

Laguna, M., Marti, R., and Campos, V. (1999). Intensification and Diversification with Elite Tabu Search Solutions for the Linear Ordering Problem. *Computers & Operations Research*, 22 :1217–1230.

Laguna, M. and Marti, R. (1999). GRASP and Path Relinking for 2-Layer Straight Line Crossing Minimization. *INFORMS Journal on Computing*, 11(1) :44–52.

Lemaréchal, C. (1989). Nondifferentiable Optimization. In Nemhauser, G.L., Rinnoy Kan, A.H.G., and Todd, M.J., editors, *Optimization*, volume 1 of *Handbooks in Operations Research and Management Science*, pages 529–572. North-Holland, Amsterdam.

Leung, J.M.Y., Magnanti, T.L., and Singhal, V. (1990). Routing in Point-to-Point Delivery Systems : Formulations and Solutions Heuristics. *Transportation Science*, 24(4) :245–260.

Magnanti, T.L., Mirchandani, P., and Vachani, R. (1993). The Convex Hull of Two Core Capacitated Network Design Problems. *Mathematical Programming*, 60 :233–250.

Magnanti, T.L., Mirchandani, P., and Vachani, R. (1995). Modeling and Solving the Two-Facility Capacitated Network Loading Problem. *Operations Research*, 43 :142–157.

Magnanti, T.L. and Mirchandani, P. (1993). Shortest Paths, Single Origine-Destination Network Desgin, and Associated Polyhedra. *Networks*, 23 :103–121.

Magnanti, T.L. and Wong, R.T. (1984). Network Design and Transportation Planning : Models and Algorithms. *Transportation Science*, 18(1) :1–55.

Minoux, M. (1989). Network Synthesis and Optimum Network Design Problems : Models, Solution Methods and Applications. *Networks*, 19 :313–360.

Mooney, E.L. and Rardin R.L. (1992). Tabu Search for a Class of Scheduling Problems. *Annals of Operations Research*, 41 :253–278.

Powell, W.B. (1986). A Local Improvement Heuristic for the Design of Less-than-Truckload Motor Carrier Networks. *Transportation Science*, 20(4) :246–357.

Ribeiro, C.C., Uchoa, E., and Werneck, R (2002). A Hybrid GRASP with Perturbations for the Steiner Problem in Graphs. *INFORMS Journal on Computing*, 14(3) :228–246.

Sellmann, M., Kliewer, G., and Koberstein, A. (2002). Capacitated Network Design, Cardinality Cuts and Coupled Variable Fixing Algorithms based on Lagrangian Relaxations. Publication tr-ri-02-234, University of Paderborn, Department of Mathematics and Computer Science.